Entrelacement des mécanismes d'identification et de respect de la vie privée pour la protection des contenus externalisés

Julien Lolive

UNIVERSITE
BRETAGNE
LOIRE

THÈSE / Télécom Bretagne
sous le sceau de l'Université Bretagne Loire

pour obtenir le grade de Docteur de Télécom Bretagne
En accréditation conjointe avec l'Ecole Doctorale Matisse
Mention : Informatique

présentée par
Julien Lolive
préparée dans le département Image & traitement de l'information
Laboratoire Labsticc

Entrelacement des mécanismes d'identification et de respect de la vie privée pour la protection des contenus externalisés

Thèse soutenue le 13 mai 2016
Devant le jury composé de :

Philippe Gaborit
Professeur, Université de Limoges / président

Jean-Marc Robert
Professeur, École de Technologie Supérieure - Canada / rapporteur

Benjamin Nguyen
Professeur, Insa - Centre Val de Loire / rapporteur

Teddy Furon
Chargé de recherche, Inria – Rennes / examinateur

Sébastien Gambs
Professeur, Université du Québec - Canada / co-directeur de thèse

Caroline Fontaine
Chargée de rercherche, CNRS-Télécom-Bretagne / Directrice de thèse

N° d'ordre : 2016telb0400

Sous le sceau de l'Université Bretagne Loire

Télécom Bretagne

En accréditation conjointe avec l'Ecole Doctorale Matisse

Ecole Doctorale – MATISSE

Entrelacement des mécanismes d'identification et de respect de la vie privée pour la protection des contenus externalisés

---

Thèse de Doctorat

Mention : Informatique

Présentée par **Julien Lolive**

Département : ITI

Laboratoire : LabSTICC Pôle : CID
IRISA Dpt : SLE

Directeurs de thèse : Caroline Fontaine et Sébastien Gambs

Soutenue le 13 mai 2016

Jury :

M. Benjamin Nguyen – Professeur à l'INSA Centre-Val de Loire (Rapporteur)
M. Jean-Marc Robert – Professeur à l'École de Technologie Supérieure de Montréal (Rapporteur)
Mme. Caroline Fontaine – Chargée de recherche au CNRS (Directrice de thèse)
M. Sébastien Gambs – Professeur à l'Université du Québec à Montréal (Co-directeur de thèse)
M. Teddy Furon – Chargé de Recherche à l'Inria Rennes Bretagne-Atlantique (Examinateur)
M. Philippe Gaborit – Professeur à l'Université de Limoges (Examinateur)

L'adversaire d'une vraie liberté est un désir excessif de sécurité.
Jean De La Fontaine

La théorie, c'est quand on sait tout et que rien ne fonctionne. La pratique, c'est quand tout fonctionne et que personne ne sait pourquoi. Ici, nous avons réuni théorie et pratique : Rien ne fonctionne... et personne ne sait pourquoi !
Albert Einstein

L'imagination est plus importante que la connaissance, car la connaissance est limitée, tandis que l'imagination englobe le monde entier.
Albert Einstein

Compter sur le gouvernement pour protéger la vie privée, c'est comme demander à un voyeur d'installer vos stores.
John Perry Barlow

Remerciements

Je tiens tout d'abord à remercier les membres du jury d'avoir bien voulu juger mes travaux. En particulier, je remercie chaleureusement Jean-Marc Robert, et Benjamin Nguyen, d'avoir accepté la charge de rapporter mon travail de thèse et effectuer des commentaires sur mon manuscrit. Je remercie également Philippe Gaborit, de m'avoir fait l'honneur de présider mon jury de thèse. Merci également à Teddy Furon d'avoir examiner mes travaux et d'avoir participé à mon jury.

Je tiens à vivement remercier Caroline Fontaine et Sébastien Gambs qui ont encadré mes travaux de recherche ainsi que pour leurs conseils avisés. Je les remercie aussi pour m'avoir donné l'opportunité de faire une thèse sous leur direction. Ils m'ont incité à être créatif, autonome et aussi à être plus rigoureux. J'ai aussi pris un grand plaisir à travailler avec eux. Pour tout cela je les remercie.

Ce travail de recherche a été réalisé au sein des équipes SFIIS (LabSTIC, Télécom-Bretagne) et CIDRE (IRISA) sur le projet POSEIDON. Je remercie les doctorants et professeurs de ces deux équipes pour les moments de détente ainsi que toutes ces discussions plus ou moins sérieuses. Je remercie aussi personnellement Cristina Onete et Julien Shreck avec qui j'ai travaillé pour leur conseil et leur expérience. Je tiens aussi à remercier les doctorants et ingénieurs de l'équipe CIDRE, Floriant, Laurent, David, Mounir, Simon, Christopher, Damien, Solenn, Mohammed, Antoine ainsi que les professeurs et maître de conférence Nico, Fred, Eric, Christophe, Ludo, Valérie et tout les autres pour tous les bons moments passés en leur compagnie.

Pour terminer, je remercie vivement mon frère Damien pour m'avoir apporté ses conseils ainsi que toute ma famille pour m'avoir soutenu, avoir cru en moi et m'avoir encouragé tout le long de mes travaux.

Résumé

Depuis la démocratisation d'Internet, la diffusion de contenus numériques protégés par des droits d'auteur (films, photos, ...) s'est accrue de manière importante, en particulier avec les réseaux pair-à-pair qui permettent de partager facilement des contenus. D'un côté, cette situation pose problème aux ayants droit de ces contenus qui veulent contrôler leur diffusion et dissuader les potentiels contrevenants. De l'autre côté, le traçage constant et généralisé des utilisateurs par des entreprises ou des gouvernements conduit ces utilisateurs à devenir méfiants et provoque une crise de confiance.

L'objectif principal de cette thèse est de pouvoir tracer les utilisateurs malveillants ayant redistribué un contenu illégalement tout en préservant la vie privée des utilisateurs honnêtes. Pour atteindre cet objectif, nous avons conçu un *protocole de personnalisation de contenus anonyme* qui modifie un contenu avant sa diffusion avec une empreinte unique, tout en garantissant aux utilisateurs honnêtes que leur vie privée est protégée. Cette empreinte permet de tracer des traîtres ayant rediffusé illégalement un contenu et provient d'un code anti-collusion permettant de tracer les traîtres en cas de collusion. Plus précisément, ce protocole est le premier de la littérature à fournir des propriétés fortes de sécurité et de respect de la vie privée se basant sur les codes anti-collusion de Tardos. Les propriétés atteintes par ce protocole sont : traçage de traîtres, *anti-framing*, anonymat révocable, non-chaînabilité des acheteurs et des contenus distribués. Ce protocole combine différentes briques de constructions telles que le transfert équivoque, les signatures de groupe et un canal de communication anonyme. Il est le premier de la littérature à allier des propriétés fortes d'anonymat avec la performance des codes anti-collusion de Tardos.

Resume

Since the democratization of Internet, the illegal redistribution of copyrighted content (movie, picture, ...) has grown tremendously, in particular to the possibility of sharing easily this type of content on P2P network. On the one hand, this situation is problematic for the owner of the content as they want to control the distribution of such data and deter potential offenders. On the other hand, the constant and generalizing tracking of users by companies and governments has lead to a trust crisis.

The main objective of this thesis is to be able to trace malicious users who redistribute illegally a content while protecting their privacy if they behave honestly. To reach this objective, we have designed an *anonymous fingerprinting protocol* enabling the personalization of a content before his distribution with a unique codeword. This codeword can be used to trace malicious users of they redistribute illegally a content. The codeword comes from an anti-collusion Tardos code that allow to trace malicious users even if the forgery is obtained by a collusion.This protocol reaches strong security and privacy properties such as traitor tracing, anti-framing, revocable anonymity, buyer and item unlinkability. To achieve these properties, the protocol combines building block such as oblivious transfer, group signatures and an anonymous communication channel. This protocol is the first one in the literature to reach strong anonymity properties as well as the performance of Tardos anti-collusion codes.

Table des matières

Liste des figures

Liste des tableaux

Listes des abréviations et symboles

τ Le WORM (Write Once Read Many ou Écrire Une fois Lire Plusieurs fois
CA Autorité de certification
$\mathtt{Cert}$ Algorithme de certification
Enc Résultat du chiffrement
$\mathsf{NIZK\text{-}PK}$ Preuve à divulgation nulle de connaissances non interactive
OA Autorité de levée d'anonymat
X Matrice contenant les empreintes des utilisateurs
$\mathsf{X_i}$ Un bit de l'empreinte d'un utilisateur
$\mathsf{ZK\text{-}PK}$ Preuve à divulgation nulle de connaissances
Z_{half} Seuil d'accusation lorsque cette dernière est réalisée sur le halfword
$\mathtt{CSign}$ Algorithme de signature d'un schéma de certification
$\mathtt{CVf}$ Algorithme de vérification de certificat d'un schéma de certification
$\mathtt{Dec}$ Algorithme de déchiffrement
$\mathtt{Insert}$ Algorithme d'insertion : cette algorithme permet d'insérer l'empreinte dans un contenu
$\mathtt{Enc}$ Algorithme de chiffrement
$\mathtt{Decode}$ Algorithme permettant de décoder (d'extraire) l'empreinte d'un contenu
$\mathtt{Fing}$ Algorithme de personnalisation
$\mathtt{AsymKeyGen}$ Algorithme de génération de clefs asymétrique
$\mathtt{HEKGen}$ Algorithme de génération de clef de chiffrement homomorphique
$\mathtt{Identify}$ Algorithme d'identification
$\mathtt{KGen}()$ Algorithme de génération de clef symétrique
$\mathtt{Sign}$ Algorithme de signature
$\mathtt{Verify}$ Algorithme de vérification de signature
χ L'alphabet utilisé pour générer une empreinte
pks Clef publique de signature
sks Clef secrète de signature
p Une probabilité du vecteur de probabilité $\mathbf{p}$
B Un acheteur
$\mathsf{Cert}()$ Certificat
$|\chi|$ Taille de l'alphabet utilisé pour générer une empreinte de personnalisation
com Mise en gage
$\mathcal{D}$ Une base de données
Dec Résultat du déchiffrement
$code()$ Algorithme d'encodage
ϵ Probabilité d'accuser un innocent
fb'_I Bloc marqué
$\mathsf{FI}^*_{\mathsf{B}^*,\mathsf{I}}$ Copie pirate
$\mathsf{FI}'_{\mathsf{I}}$ Contenu personnalisé
W' Algorithme de tatouage de contenu
f Empreinte de personnalisation insérée dans le contenu distribué
gsk Clef privée de signature de groupe
hbc Honnête mais curieux
hpk Clef publique de chiffrement homomorphe
hsk Clef secrète de chiffrement homomorphe

Hw	Le halfword obtenu par le vendeur durant la vente
id	Identifiant du contenu
J	Le juge
k	Une clef
κ_I	Matrice de clefs de chiffrement spécifique à un contenu
m	Taille de l'empreinte
$\mathcal{LI}$	Liste des identités après ouverture de $\mathcal{LO}$
$\mathcal{LO}$	Liste d'ouverture, cette liste contient les signatures qui seront ouvertes par l'autorité de levée d'anonymat
$\mathcal{LS}$	Liste de suspects générée par le vendeur au moment de l'accusation
mal	Malveillant
M	Un vendeur
id	Identifiant de l'utilisateur
n	Nombre d'acheteur
c	Nombre de membres effectuant une collusion
N	Paramètre de quantification permettant de quantifier les p_i
ℓ	Nombre de contenus que possède le vendeur
$open$	Ouverture de mise en gage
OT	k-parmi-a Transfer Équivoque (ou Oblivious Transfer en anglais) : OT_k^a
$\Pi()$	Permutation
δ	Probabilité d'accuser un innocent
π	Une preuve
I	Le contenu original
pk	Clef publique
$\mathsf{pk}_{\mathbb{G}}$	Clef publique permettant de vérifier les signatures de groupe
δ_{max}	Probabilité d'accuser un innocent maximum autorisé par le système
ppar	Paramètres publics
$record_B = ()$	Enregistrements que possèdent l'acheteur
$record\_list$	Liste d'enregistrements
$record_M = ()$	Enregistrements que possèdent le vendeur
rnd	Nombre aléatoire
R	Un nombre aléatoire (nonce)
S	Score d'accusation
sk	Clefs secrète
spar	Paramètres secrets
swk	Clef secrète de tatouage
Z	Seuil total d'accusation
TS	Étiquette temporelle
t	Numéro de transaction
σ_{clef}	La signature a été générée avec la clef $clef$
σ_I^B	La signature concerne le contenu I et est générée par l'utilisateur B
$text$	Contrat décrivant la vente
$\mathbf{p}$	Vecteur de probabilité
WI	Contenu tatoué
f_*	Un bit de l'empreinte extrait d'une copie pirate

Introduction

Depuis la démocratisation d'Internet auprès du grand public, la diffusion de contenus numériques protégés par des droits d'auteur (par exemple, films, musiques, photos, bases de données, ...) s'est accrue de manière importante et d'autant plus que les réseaux pair-à-pair permettent de partager des contenus facilement et rapidement. Cette situation pose problème aux ayants droit de ces contenus qui veulent contrôler la diffusion de ceux-ci. Techniquement, il est très difficile d'empêcher une personne malveillante de distribuer un contenu acquis légalement et soumis au droit d'auteur. Par contre, il est possible de l'en dissuader.

Pour atteindre cet objectif, des protocoles de *personnalisation de contenu* (*fingerprinting* en anglais) peuvent être utilisés. Ceux-ci permettent de personnaliser un contenu avant sa diffusion avec une empreinte unique, insérée (Figure 0.2) à l'aide d'une technique de tatouage de contenu robuste (*watermarking* en anglais). Cette empreinte permet de tracer des traîtres [1] ayant rediffusé illégalement un contenu. En d'autres termes, la personnalisation de contenu consiste à attacher à un contenu une empreinte unique liée à un utilisateur spécifique. Le contenu de cette marque peut être de différentes natures : l'identité de l'utilisateur récupérant le contenu de manière légale, un mot de code provenant d'un code anti-collusion, une suite de bits aléatoire, etc Cependant, pour contrer des collusions d'utilisateurs, il faut que l'empreinte insérée dans le contenu ait une structure spécifique, d'où le concept de *code anti-collusion.*

Définition 0.1. (Collusion): *Informellement, une* collusion *est un ensemble d'utilisateurs qui mettent en commun leur copie d'un contenu obtenu légalement dans le but de fabriquer une copie pirate ne pointant vers aucun des membres de la collusion.*

La personnalisation de contenu ressemble beaucoup à la *stéganographie* [38] (Figure 0.1) ou encore au *tatouage de contenu* [71] (Figures 0.2 et 0.3). En effet, ces trois techniques font partie du domaine de la *dissimulation de l'information* (*information hiding* en anglais) [71]. Les différences entre ces trois techniques résident dans l'objectif et les propriétés de sécurité requises. La *stéganographie* permet de rendre une communication furtive en cachant un message à envoyer dans un support (par exemple, une photo). L'*imperceptibilité* de ce message doit donc être totale, peu importe les moyens, informatique et humain, utilisés pour en détecter la présence. Pour le *tatouage*, bien que la marque puisse être imperceptible ou visible, elle doit surtout être robuste ou fragile suivant l'utilisation que l'on veut faire du tatouage. La robustesse d'un tatouage se réfère au fait que la marque ne doit pas pouvoir être supprimée alors que la fragilité est

1. Un traître se définit comme une personne ayant acquis un contenu légalement et le distribuant sans autorisation, après la réalisation d'une attaque ayant pour but d'altérer, de supprimer l'empreinte en effectuant une collusion (Définition 0.1) ou non [133].

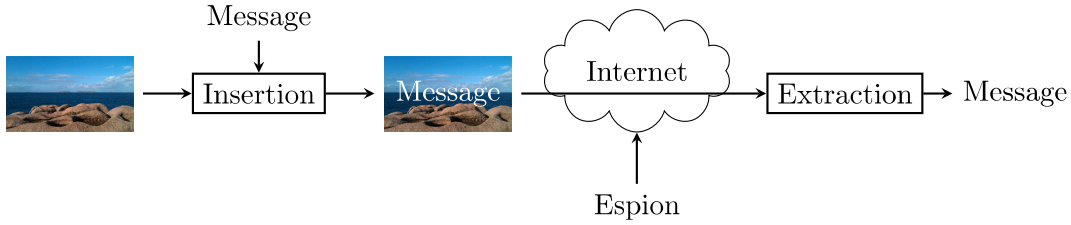

FIGURE 0.1.: Exemple d'utilisation de la stéganographie.

FIGURE 0.2.: Insertion d'une marque.

FIGURE 0.3.: Extraction d'une marque.

le fait qu'une modification du contenu se répercute sur la marque. De plus, le tatouage de contenu permet de dissimuler (si besoin d'imperceptibilité) une information dans un contenu, information qui peut être utilisée pour le droit d'auteur (par exemple, le message peut être l'identité du propriétaire, inséré à l'aide d'un tatouage robuste) et pour la vérification de l'intégrité d'un contenu (tatouage fragile). Un tatouage visible peut être utilisé pour diffuser un contenu dans le but de montrer son travail (par exemple, photographe). De surcroît, il n'y a qu'une copie marquée contenant le message qui est distribuée à tous les utilisateurs. Enfin, la *personnalisation de contenu*, quant à elle, intervient pour cacher une information telle que l'identité de la personne récupérant le contenu ou encore un mot de code, dans le but d'avoir un contenu personnalisé avant sa distribution. Cette empreinte a pour but le traçage de traîtres en cas de redistribution illégale d'un contenu, et est insérée à l'aide d'une technique de tatouage pour combiner les propriétés de robustesse et sécurité du tatouage et celles du traçage de traîtres.

La figure 0.4 illustre l'utilisation d'un protocole de personnalisation de contenu. Le but d'un tel protocole est de personnaliser le contenu récupéré par le client à l'aide d'un mot de code provenant d'un code anti-collusion. Dans ce cadre, le vendeur doit être capable de retrouver avec une forte probabilité *au moins* un des membres de la collusion (c'est-à-dire, de tracer un traître).

Par ailleurs, la protection de la vie privée des utilisateurs est devenue un enjeu de taille au vu des nombreux scandales de ces dernières années. En effet, de nombreuses entreprises tracent les utilisateurs sur Internet (comme Google à l'aide de Doubleclick) ainsi que des agences gouvernementales (PRISM pour la NSA ou la loi de renseignement en France). Ce traçage généralisé conduit les utilisateurs d'Internet à devenir de plus en plus méfiants et provoque une crise de confiance. Ainsi, ils regardent de plus en plus ce qu'ils partagent avec le monde et essaient de mieux protéger leur vie privée. Il y a donc une réelle aspiration à la protection de la vie privée dans notre société. Pour l'instant,

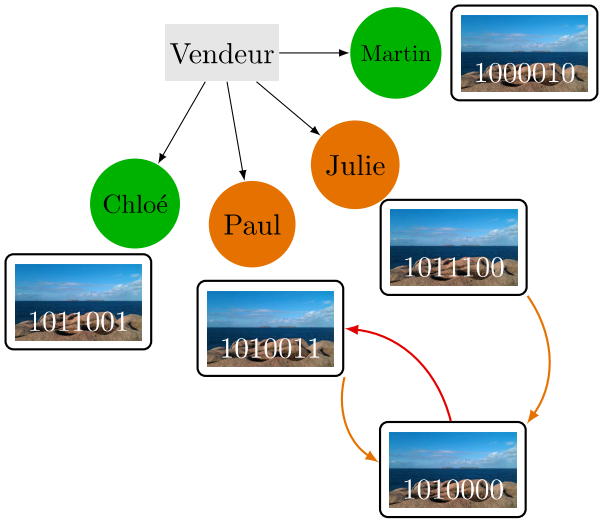

FIGURE 0.4.: Exemple de protocole de personnalisation de contenus. Le vendeur est le fournisseur de contenu. Les acheteurs (Martin, Julie, Paul et Chloé) récupèrent un contenu auprès du vendeur. Dans ce cas, ces quatre utilisateurs récupèrent une copie du contenu qui est marquée avec une empreinte unique liée à leur identité (resp. 1000010,1011100,1010011,1011001) dans le but de personnaliser le contenu. Paul et Julie réalisent une collusion (en orange) dans le but de fabriquer une copie pirate contenant une empreinte qui ne peut être reliée à aucun des deux (ici, l'empreinte de la copie pirate est 1010000). Pour contrer cela, le vendeur doit être capable de retrouver, avec une forte probabilité, au moins un des membres de la collusion (flèche rouge).

des textes légaux (par exemple, Directive européenne 95/46/EC[5]) encadrent le droit au respect de la vie privée tandis que des entités comme les autorités de protection des données, telle que la CNIL (Commission nationale de l'informatique et des libertés), sont garantes de l'application de ces textes (par exemple, amende à Google). Cependant, ceci n'est pas suffisant pour empêcher la collecte d'informations sur les utilisateurs. De plus, il est difficile de vérifier si les textes de loi sont appliqués, malgré l'existence des autorités de protection des données, ainsi que l'usage réel des données collectées. Par exemple, si une entreprise collecte des données sur ses usagers en les prévenant par des conditions d'utilisation à valider, elle respecte les textes de loi. Cependant, l'utilisation réelle des données par cette entreprise peut différer de ces conditions et cela est très difficilement vérifiable. Pour aller plus loin, il faut agir en trouvant un moyen de fournir les mêmes services aux utilisateurs tout en garantissant le droit à leur vie privée de par la conception du service. Cependant, bien que la vie privée des utilisateurs doive être protégée, il existe des situations réalistes où il est nécessaire aussi de pouvoir identifier certaines personnes commettant des actes malveillants. Dans ces situations, il est donc nécessaire de réussir à concilier ces deux aspects (sécurité et respect de la vie privée).

Deux grandes familles d'approches existent pour protéger le respect de la vie privée : la protection *a priori* et la protection *a posteriori*. Le respect de la vie privée *a priori*,

c'est-à-dire en amont, est le fait de garantir le respect de la vie privée dès la conception du système et avant son déploiement. À l'inverse dans la protection *a posteriori*, le respect de la vie privée n'est pas garanti de manière absolue, et ne tient que jusqu'à ce qu'une brèche soit détectée après le déploiement du système. La réaction se fait donc en aval. Dans cette thèse, nous nous plaçons dans le cas *a posteriori*, c'est-à-dire que nous garantissons le respect de la vie privée des utilisateurs tant que ceux-ci n'agissent pas de manière malveillante. Autrement dit, un utilisateur honnête reçoit des garanties de respect de la vie privée fortes alors que l'identité d'un utilisateur malveillant pourra être retrouvée.

Le traçage des utilisateurs sur Internet est différent du traçage de traîtres dans le cadre de la personnalisation de contenus. Le premier est un acte malveillant, brisant la vie privée, qui consiste à suivre les faits et gestes des utilisateurs en ligne. On veut donc protéger les utilisateurs contre ce type de traçage pour garantir le droit au respect de la vie privée. Le traçage de traîtres est quant à lui totalement légitime. En effet, les traîtres sont des utilisateurs qui, dans notre contexte, distribuent illégalement un contenu sans l'accord de l'ayant droit. Il s'agit donc d'une propriété de sécurité que nos protocoles doivent impérativement respecter. Dans cette thèse, lorsque nous parlerons de *traçage* nous ferons référence aux traçages des utilisateurs sur Internet alors que nous utiliserons le terme *traçage de traîtres* pour la protection de contenu.

Les travaux de recherche de cette thèse, présentés dans ce mémoire, se situent au croisement de différents domaines tels que la cryptographie, la personnalisation de contenu, la personnalisation de bases de données et le respect de la vie privée. L'objectif poursuivi est l'entrelacement des mécanismes d'identification et de protection de la vie privée. Plus particulièrement, ces travaux ont porté sur :

1. Conception d'un protocole de *personnalisation de contenu* asymétrique, basé sur les codes anti-collusion de Tardos et respectant la vie privée des utilisateurs. Ce protocole permet, grâce aux codes de Tardos, de dissuader des utilisateurs malveillants de redistribuer, illégalement, des contenus (par exemple, photo, vidéo, ...) et de pouvoir tracer les traîtres. De plus, nous garantissons qu'un acheteur innocent ne peut pas être faussement accusé par un vendeur malveillant ou par une collusion d'acheteurs. Aussi, la vie privée des acheteurs est respectée et nous empêchons le profilage de ceux-ci en cachant ce qu'ils achètent. Cependant, même si le fait de protéger la vie privée des utilisateurs est primordial, il est essentiel que le vendeur puisse retrouver une personne malveillante. Ainsi, l'anonymat des acheteurs doit donc être révocable. Nous proposons pour ce protocole un modèle de sécurité, des preuves formelles ainsi qu'une implémentation permettant de réaliser des tests pratiques.
2. Sur la personnalisation de bases de données anonymisées. Dans une troisième contribution, nous présenterons une méthode combinant la personnalisation et l'assainissement de bases de données. L'objectif de cette contribution est différent des deux précédentes. En effet, dans la première contribution, ce sont les acheteurs qui sont anonymes (tout comme dans la deuxième) et les articles vendus ne sont pas connus du vendeur. Dans cette troisième contribution, ce ne sont pas

les acheteurs, mais les informations personnelles d'utilisateurs présentes dans la base de données qui sont assainies (par exemple, numéro de sécurité sociale, nom, prénom,...). Ceci est nécessaire lorsque, par exemple, la base de données délivrée à une tierce partie est une base de données médicale, qui contient des informations personnelles et sensibles liées à la santé d'un patient. De plus, nous voulons que cette base de données soit personnalisée dans le but de tracer les acheteurs malveillants qui redistribueraient la base de données illégalement.

Le plan global dans cette thèse est le suivant. Nous allons nous intéresser, dans une première partie à l'état de l'art. Dans le Chapitre 1 nous présentons le traçage de traîtres et la personnalisation de contenus (Section 1.3). De plus, nous survolerons les concepts des codes anti-collusion, et plus précisément les codes de Tardos (Section 1.5). La section 1.6.2 est consacrée à la personnalisation de contenus où l'information insérée est un mot de code provenant de codes anti-collusion. Nous présentons, à la fin de ce chapitre, la personnalisation de contenus spécifique aux bases de données. Ensuite, nous nous intéressons, dans le Chapitre 2 à la protection de la vie privée avant de présenter l'état de l'art des protocoles de personnalisation de contenu préservant celle-ci dans la Section 2.4. Puis, nous présentons l'état de l'art des solutions de la littérature concernant la personnalisation de base de données assainie. Par la suite, dans une seconde partie, nous présenterons nos contributions. Ainsi, dans le chapitre 3 nous décrivons nos contributions sur la conception de deux protocoles de personnalisation de contenus respectant la vie privée, nommés PIMENTO et PIMENTO+. Nous conclurons ce mémoire de thèse en résumant les travaux effectués et en proposant des pistes de recherche améliorant les travaux présentés ici ou bien ouvrant de nouvelles directions.

Première partie .

État de l’art

1. Personnalisation de contenus et traçage de traîtres

Dans ce chapitre, nous nous intéressons aux protocoles de personnalisation de contenus permettant de tracer les traîtres (c'est-à-dire, les acheteurs ayant redistribué un contenu numérique de manière malveillante) en utilisant une technique de tatouage. Nous présenterons aussi les différents types d'attaques existantes sur de tels protocoles. Nous ne présenterons pas les différentes méthodes de tatouage de contenu en détail dans ce mémoire, bien que nous présentions les différents types de tatouage existants.

1.1. Tatouage de contenu

Le tatouage de contenu numérique (ou *watermarking* en anglais) permet d'insérer un tatouage dans un contenu de manière imperceptible à l'œil humain [43]. Le but étant de générer une copie tatouée avant de la distribuer pour connaître, par exemple, l'identité du propriétaire du contenu (dans le cas où le tatouage est relié à son identité) ou encore pour s'assurer que les données n'ont pas été modifiées (on parle alors d'intégrité des données). Dans cette dernière optique, la même copie est fournie à tous les utilisateurs. En d'autres termes, il n'y a pas de personnalisation de contenus dans le sens où la copie fournie à l'utilisateur est commune à tous les acheteurs. De plus, le tatouage doit être invisible (c'est-à-dire, que ce dernier n'est pas visible par l'oeil humain).

Deux algorithmes sont nécessaires(Figure 1.1 et 1.2) :

— `Insert` : Cet algorithme fait appel à une fonction W qui permet d'insérer une marque dans le contenu en prenant en entrée le contenu I, la marque f de taille m et une clef secrète sk qui permet de calculer les positions d'insertion accueillant les bits $f^{i,j}$ et fournit en sortie un contenu tatoué WI.

— `Decode` : Cet algorithme extrait (décode) la marque présente dans un contenu. Plus formellement, prenant un contenu tatoué WI, la clef secrète d'insertion sk, cet algorithme fournit en sortie f_*. Cette dernière possède potentiellement des erreurs ou des effacements résultant d'attaques volontaires ou non sur le contenu.

1.1.1. Propriétés de tatouage

Les différentes propriétés d'un schéma de tatouage sont les suivantes :

— *Imperceptibilité du tatouage :* Le tatouage est dit imperceptible lorsqu'une fois la marque insérée, celle-ci n'empêche pas une utilisation correcte du contenu (c'est-à-dire, la marque n'est pas visible à l'œil nu).

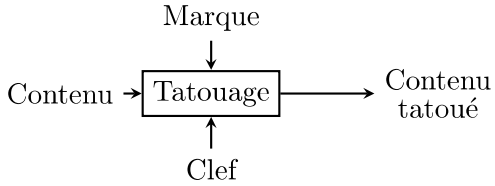

FIGURE 1.1.: L'algorithme de tatouage insère la marque f fournie en paramètre de l'algorithme W dans le contenu I en utilisant la clef secrète sk qui permet de calculer les emplacements qui recevront la marque.

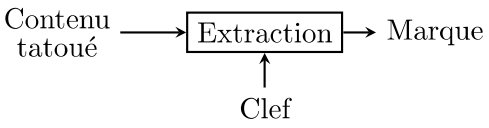

FIGURE 1.2.: En utilisant la clef secrète sk et le contenu tatoué WI, l'algorithme de décodage extrait (décode) la marque f_* pour identifier, par exemple, le propriétaire du contenu.

— *Tatouage aveugle ou non aveugle :* Le tatouage aveugle signifie que lors du décodage du tatouage le contenu original n'est pas nécessaire, contrairement au tatouage non aveugle.
— *Symétrique ou Asymétrique :* Un schéma de tatouage est dit symétrique si la clef d'insertion et d'extraction est la même, contrairement au tatouage asymétrique [2].
— *Tatouage robuste ou fragile :* Le tatouage fragile permet de vérifier l'intégrité visuelle du contenu tatoué. En effet, le but du tatouage fragile est d'insérer un tatouage dans le contenu qui s'en retrouvera modifié en cas d'altération par l'utilisateur. Au contraire d'un tatouage fragile, un tatouage robuste est nécessaire lorsque la marque insérée ne doit pas être détruite ou altérée par des modifications du contenu. Le tatouage robuste est utilisé notamment dans le traçage de traîtres, la preuve de propriété ou d'appartenance (c'est-à-dire, la marque insérée permet de prouver que l'on est bien le propriétaire du contenu), etc. Un utilisateur peut modifier la marque insérée, volontairement [3] ou non [4], en réalisant différentes attaques. Bien entendu, plus le tatouage est robuste moins il sera sensible aux attaques. Un tatouage robuste doit pouvoir résister aux attaques suivantes (liste non exhaustive fournie à titre d'exemple) [5] :
  — *Transformations géométriques :* Le but de l'attaque est de modifier l'image par une rotation ou encore un redimensionnement.
  — *Conversion analogique/numérique :* L'objectif de l'attaque est par exemple,

2. Les tatouages symétriques et asymétriques peuvent se rapprocher du chiffrement symétrique où la clef de chiffrement et de déchiffrement est identique et du chiffrement asymétrique où elles sont différentes. Il ne faut pas les confondre avec les schémas de personnalisation de contenus symétriques et les protocoles asymétriques que nous présentons dans la Section 1.6

3. Le but de l'adversaire est de supprimer ou de suffisamment altérer la marque, par exemple pour distribuer illégalement un contenu sans être tracé (c'est-à-dire, sans que l'on remonte jusqu'à lui)

4. L'utilisateur peut vouloir modifier par exemple le sens d'une image, sa compression, ... Ces « attaques » n'ont pas pour but de supprimer la marque

5. Pour plus de détails sur ces attaques vous pouvez vous reporter aux cours en ligne de Caroline Fontaine qui se trouve à l'adresse `http://www.picsi.org/parcours_17_90.html` – Accès le 16 février 2016

pour une image, de réaliser une impression suivie de la numérisation de cette dernière. Pour un film, l'attaque serait par exemple d'enregistrer un film dans une salle de cinéma avec un caméscope.

— *Compression :* La compression d'une image fait perdre des informations, rendant le poids de l'image inférieur à celui de l'original. Il est possible qu'une partie de ces informations perdues fassent partie du tatouage lui même.

Dans ce mémoire, nous ne nous intéressons qu'au tatouage robuste. De plus, nous ne décrirons pas les techniques de tatouage de contenus bien que des techniques de tatouage robustes existent, par exemple [134]. En effet, le tatouage de contenu n'est pas mon sujet et mes contributions ne sont pas sur ces méthodes.

1.2. Traçage de traîtres

Objectif. Tout d'abord, un traître est un utilisateur qui, ayant acquis une copie d'un contenu numérique de manière légale, la diffuse illégalement. Le traçage de traîtres est le fait de retrouver une personne malveillante, nommée *traître*. Pour cela, il faut utiliser la personnalisation de contenu qui permet au propriétaire de distribuer un contenu personnalisé, c'est-à-dire unique, à des personnes autorisées après avoir personnalisé celui-ci (par exemple, avec le nom de la personne recevant le contenu ou un mot d'un code anti-collusion) et de retrouver ce traître. En d'autres termes, le contenu numérique fourni par le propriétaire est modifié avant la distribution et la différence entre la copie du contenu et le contenu original est la personnalisation de celui-ci. L'empreinte insérée pour un mot de code de Tardos binaire est par exemple de la forme : 00110101. Cette personnalisation peut être vue comme le fait d'insérer des erreurs dans le contenu original, ce qui rend le contenu vendu unique.

Traçabilités : Faible et Forte. Il existe deux types de traçabilités : *traçabilité faible* et *traçabilité forte*. La traçabilité faible permet de retrouver l'un des traîtres avec une forte probabilité et donc possède aussi une faible probabilité de non-détection (des traîtres peuvent passer entre les mailles du filet). Cependant, les utilisateurs accusés ont une forte probabilité d'être coupables. L'inconvénient est que des innocents peuvent être accusés (fausse alarme), mais la probabilité que cela arrive est très faible. Ces deux probabilités sont maîtrisées et respectées.

Contrairement à la traçabilité faible, la traçabilité forte est beaucoup plus contraignante. En effet, celle-ci ne tolère aucune erreur d'accusation, ce qui rend les mots de codes beaucoup plus longs [54] et l'alphabet utilisé plus important $\chi = \{0...q\}$. Par conséquent, leur insertion invisible dans un contenu est beaucoup plus difficile. De plus, le temps d'insertion de l'empreinte dans le contenu dépend, entre autres, de la longueur de celle-ci, ce qui fait que plus l'empreinte est longue plus le temps pour l'insérer s'accroît. Dans une situation de production, ceci peut s'avérer handicapant.

Dans notre cas d'étude, nous utilisons la traçabilité faible qui est la seule utilisable en pratique compte tenu des performances des algorithmes de tatouage robuste.

1.3. Personnalisation de contenus

La personnalisation de contenus est une technique vieille de plusieurs siècles, mais étudiée que depuis les années 80 pour les contenus numériques. Nous allons voir dans cette section l'objectif de la personnalisation de contenus ainsi que les différents types de protocoles existants dans la littérature.

1.3.1. Objectif de la personnalisation de contenus

La *personnalisation de contenu* (ou *fingerprinting en anglais*) est le fait d'insérer une empreinte personnalisée dans un contenu à l'aide d'une technique de tatouage afin de tracer les traîtres à l'aide de l'empreinte. La personnalisation de contenus fut introduite pour la première fois par Wagner en 1983 [133]. Cependant, celle-ci fut utilisée, il y a plusieurs siècles, par Neper qui modifia ses tables de logarithmes en changeant des décimales non significatives. Ces décimales étaient changées pour chaque personne à qui ces tables de logarithmes étaient distribuées. Cette méthode lui permettait de tracer les traîtres ayant illégalement (c'est-à-dire, sans son autorisation) publié une copie de ses tables, tout en conservant la pertinence et l'utilité de celles-ci. De nos jours, l'empreinte est insérée dans un contenu numérique pour atteindre la propriété de traçage de traîtres. Cette empreinte peut correspondre au nom de l'acheteur, à une suite aléatoire de bits ou à un mot de code d'un code anti-collusion et l'empreinte doit être invisible. En effet, dans le cas où une personne achèterait, par exemple un film, l'empreinte ne doit pas être visible (Figure 1.3 [6]) pour que le contenu puisse être utilisé dans les meilleures conditions tout en assurant au propriétaire d'être capable de tracer les traîtres. Il faut noter que l'insertion d'une empreinte dans un contenu entraîne une insertion d'erreur dans celui-ci. En effet, comparé à l'original, le contenu personnalisé est modifié. De plus, l'empreinte doit être robuste à certaines attaques. Le seul moyen d'arriver à cela est d'utiliser un tatouage robuste pour insérer cette empreinte.

Dans cette thèse, nous nous intéresserons plus particulièrement aux protocoles de personnalisation de contenus qui permettent de tracer des traîtres en insérant une empreinte personnalisée à l'aide d'un protocole de tatouage de contenu (*watermarking* en anglais). Il existe trois familles de schémas/protocoles de personnalisation de contenus : *Symétrique (Figure 1.4, qui est un schéma)*, *Asymétrique (Figure 1.5)* détaillé ci-après et *Anonyme* détaillé dans le chapitre 2.4 qui sont tout les deux des protocoles. Ces schémas/protocoles de personnalisation de contenus peuvent être monétaires (c'est-à-dire que l'acheteur paie pour acquérir un contenu numérique) ou non. Même s'ils ne sont pas monétaires, nous emploierons les termes *acheteur* et *vendeur*, car ce sont ceux employés dans la littérature (*buyer* et *merchant* en anglais). De plus, nous emploierons le terme *schéma* pour la personnalisation de contenus lorsqu'il n'y a pas d'interaction entre le vendeur et l'acheteur contrairement aux protocoles. Aussi, un protocole permet de réglementer les parties sur leurs rôles et sur ce qu'elles doivent ou ne doivent pas faire.

6. L'image a été personnalisée avec le logiciel BrokenArrow disponible sur `http://bows2.ec-lille.fr/` conçu par Teddy Furon et Patrick Bas [55]. – Accès le 16 février 2016

FIGURE 1.3.: À droite, l'image originale en niveau de gris. À gauche, l'image personnalisée avec Broken Arrows. La différence entre ces deux images est que dans celle de gauche une empreinte a été insérée de manière imperceptible.

1.3.2. Symétrique et asymétrique

Entités. Dans un protocole de personnalisation de contenus, nous trouvons habituellement les entités suivantes :

— *Acheteur :* L'acheteur est une personne qui veut acquérir un contenu (film, chanson, photo, ...).
— *Vendeur :* Le vendeur est le fournisseur de contenu. En cas de redistribution illégale de la part d'un acheteur, le vendeur génère une preuve de culpabilité qui est ensuite fournie au juge. Le vendeur doit pouvoir retrouver en cas de collusion, avec une forte probabilité, au moins un des membres de la collusion.
— *Juge :* Le juge est chargé de vérifier la preuve de la culpabilité d'un acheteur suspect et si ce dernier est coupable, de son accusation.

Symétrique. Dans un *schéma de personnalisation de contenus symétrique*, le vendeur génère lui-même une empreinte personnalisée avant de l'insérer dans le contenu et de distribuer ce dernier. Par conséquent, il n'y a que le vendeur qui connaît cette empreinte. En effet, l'acheteur n'intervient à aucun moment dans cette génération.

À un niveau abstrait, un schéma symétrique fonctionne de la manière suivante. :

1. Préparation de copies sensiblement différentes (personnalisées) pour chaque acheteur. La différence entre les copies n'est autre que l'empreinte insérée.
2. Vente (ou diffusion) du contenu à l'acheteur.
3. Stockage de l'empreinte en la liant avec l'identité de l'acheteur.
4. Si le vendeur trouve une copie pirate, il peut retrouver, avec une probabilité ϵ, un coupable en comparant l'empreinte du contenu trouvé avec les empreintes qu'il stocke.

Les étapes de l'identification sont en général les suivantes.

— Le vendeur extrait l'empreinte f_* de la copie pirate.
— Il compare cette empreinte avec toutes les empreintes f_j (j étant un acheteur) présentes dans sa base d'empreintes.

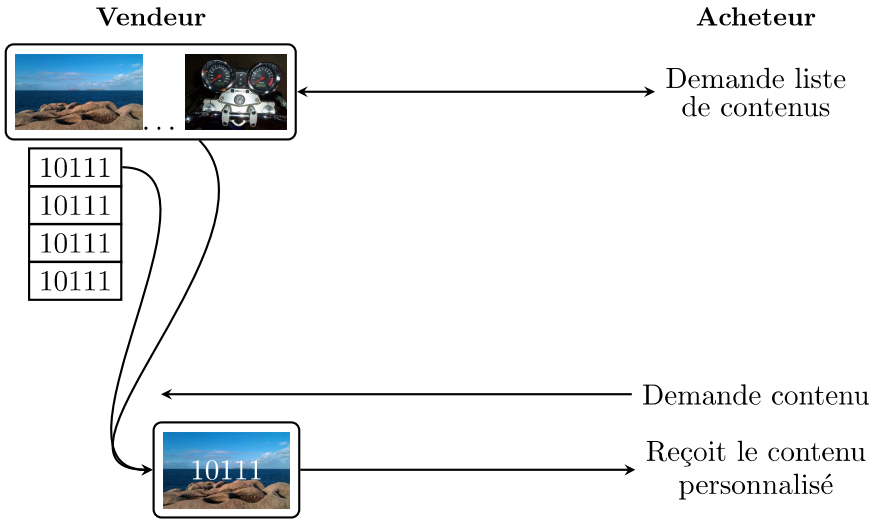

FIGURE 1.4.: Modèle général de protocole de personnalisation de contenus symétrique.

— Si une correspondance est trouvée, alors le vendeur peut suspecter un acheteur particulier et fournir une preuve de culpabilité au juge.
— Le juge, après vérification de la preuve, confirme ou infirme la culpabilité de l'acheteur.

Il est possible que l'empreinte extraite de la copie distribuée illégalement ne soit pas dans la base à cause d'une collusion ou encore parce qu'elle a été altérée. L'altération peut être due à une compression ou toutes autres attaques présentées dans la section 1.4. Pour certaines attaques, et plus précisément pour l'exemple précédent, seul la technique de tatouage permet de les contrer. Pour éviter les erreurs, il faut respecter la propriété de robustesse. Cette propriété est atteinte par le tatouage, qui doit être robuste, et par la conception du schéma. Pour finir, pour être capable de retrouver un traître en cas de collusion il faut utiliser un code anti-collusion qui possède une structure spécifique.

Asymétrique. Les défauts principaux de l'approche symétrique sont les suivants. Un vendeur malveillant peut forger une copie valide correspondant à un acheteur légitime et en conséquence l'accuser à tort. En effet, le vendeur possède les empreintes liées aux identités des acheteurs ayant récupéré ces contenus, ainsi que les contenus originaux. Un autre problème est le fait qu'un acheteur malveillant, ayant diffusé illégalement une copie d'un contenu, peut se dédouaner en clamant que le vendeur a pu effectuer la diffusion illégale et que ce dernier essaie de l'accuser. Pour pallier ces faiblesses, les *protocoles de personnalisation de contenus asymétriques* (Figure 1.5) utilisent un protocole biparti entre le vendeur et l'acheteur lors de la génération de l'empreinte [105, 107]. La conception des protocoles de personnalisation de contenus asymétriques a rapidement remplacé l'étude des schémas symétriques. Dans ce type de protocoles, l'empreinte est générée, entièrement ou en partie, par l'acheteur lui-même. À la fin du

protocole, le vendeur n'apprend pas l'empreinte complète, mais seulement une partie de cette empreinte qui est communément appelée *halfword* en anglais[7]. Cependant, en pratique, le *halfword* n'est pas forcément égal à la moitié de l'empreinte. En effet, cela dépend du fonctionnement du protocole, mais dans tous les cas il est nécessaire que le *halfword*halfword soit d'une taille suffisante pour permettre une accusation la plus précise possible. De son côté, l'acheteur reçoit le contenu personnalisé et en aucun cas le contenu original. Il faut noter, et garder en mémoire, que le vendeur ne doit pas choisir les positions composant le *halfword*. En effet, si c'était le cas il pourrait pour une empreinte f de taille m demander $\frac{m}{2}$ bits en prenant un bit tous les deux bits (c'est-à-dire, ceux pour les positions paires) pour un premier acheteur et pour un deuxième il pourrait faire la même chose, mais en prenant ceux pour les positions impaires. Dans ce cas, il pourrait forger une copie qui serait composée des *halfwords* de deux acheteurs et il pourrait les accuser tous les deux. De plus, l'acheteur ne doit pas apprendre le *halfword* révélé au vendeur, car, lors d'une collusion, il saurait quelles parties de son contenu numérique il ne doit pas utiliser. Nous verrons dans nos contributions (Chapitres 3) que nous nous sommes attachés à respecter ces conditions pour les protocoles que nous avons développés.

Dans un protocole de personnalisation de contenus asymétrique, lorsque le vendeur extrait l'empreinte d'une copie illégale, il regarde seulement les bits correspondants aux indices du *halfword*. Pour l'identification d'un potentiel acheteur malveillant, le vendeur utilise une fonction d'accusation qui fournit une preuve de la culpabilité d'un acheteur. Cette preuve est ensuite transmise au juge avec l'identité de l'acheteur. Puis, le juge peut confirmer ou infirmer la preuve et punir l'acheteur malveillant sur la base d'un examen plus complet impliquant l'ensemble de l'identifiant. Si le juge infirme la preuve, après examen complet de l'identifiant, cela peut vouloir dire que le vendeur est malveillant, car il essaie d'accuser à tort un acheteur innocent. Dans ce cas, le juge peut mettre le vendeur sur liste noire ou entreprendre toute autre action appropriée.

Généralement, un protocole de personnalisation de contenus asymétrique se déroule de la manière suivante.

1. L'acheteur génère une empreinte, à partir d'informations fournies par le vendeur ou sans aucune interaction avec le vendeur selon les cas. L'empreinte peut être une suite de bits tirés aléatoirement en fonction d'informations fournies par le vendeur, l'identité de l'acheteur ou encore un mot provenant d'un code anti-collusion.
2. Un protocole de tatouage de contenu permet d'ajouter une empreinte au contenu original sans que le vendeur ait connaissance du contenu tatoué et sans que l'acheteur connaisse le contenu original. Il est possible de le faire avec, par exemple, du chiffrement homomorphe comme dans [76] qui utilise la propriété homomorphique de l'addition du schéma de chiffrement proposé par [96]. Il existe cependant d'autres moyens que le chiffrement homomorphe que nous détaillerons plus tard dans les chapitres présentant nos contributions. En effet, suivant la structure du protocole il est possible de faire autrement que d'insérer dans le domaine chiffré.

7. En français, nous pourrions traduire *halfword* par demi-mot. Cependant, dans un souci d'esthétisme nous garderons le terme anglais.

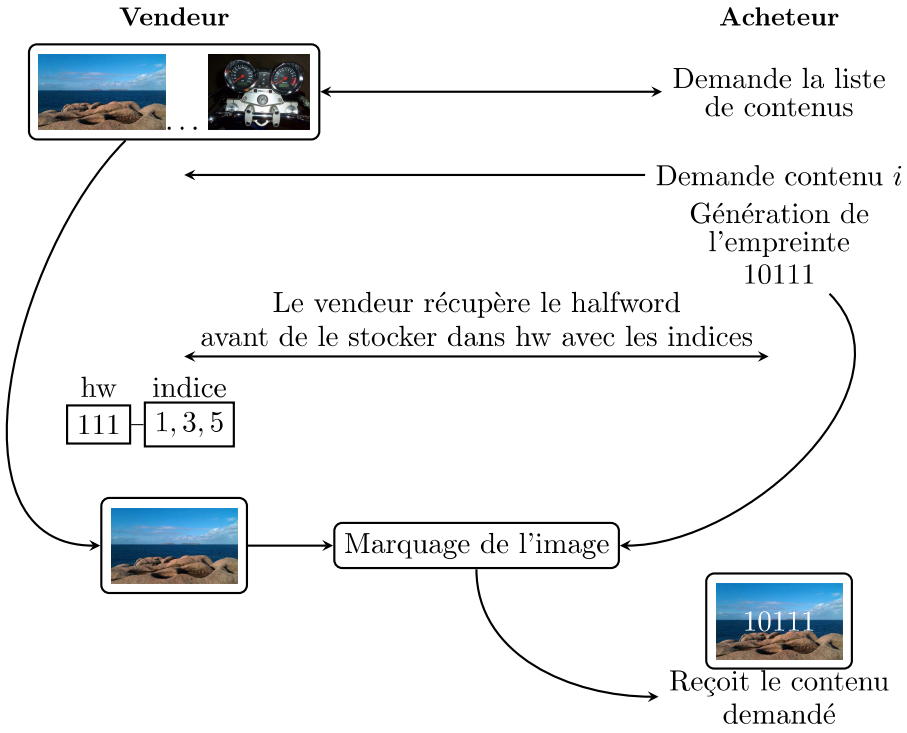

FIGURE 1.5.: Modèle général de protocole de personnalisation de contenus asymétrique

3. Le vendeur apprend le *halfword* qui est stocké et lié à l'identité de l'acheteur.
4. L'accusation par le vendeur ne s'appuie que sur le *halfword*, car il ne connaît pas l'empreinte complète.
5. Le juge valide ou invalide la preuve. S'il valide la preuve, il considère que l'acheteur est coupable.

Les premiers chercheurs à avoir conçu un tel protocole sont Pfitzmann et Schunter en 1996 [105]. Nous détaillons les protocoles [105] et [107] dans la section 1.6 de ce chapitre.

1.3.3. Propriétés de sécurité

Un schéma de personnalisation de contenus symétrique doit posséder la propriété suivante :

— *Traçage de traîtres :* Le vendeur doit être capable de retrouver l'identité des acheteurs malveillants.

En plus de cette propriété, il faut respecter la propriété de *conformité* (*correctness* en anglais) définissant que si tous les participants au protocole suivent ce dernier conformément, alors les sorties de celui-ci doivent être valides et conformes à sa spécification.

En plus des propriétés précédentes, un protocole de personnalisation de contenus asymétrique doit aussi respecter la propriété suivante :

— *Anti-framing :* Un acheteur innocent ne doit pas pouvoir être accusé à tort par un vendeur malveillant ou par une collusion d'acheteurs malveillants.

1.4. Attaques sur les protocoles de personnalisation de contenus

Il existe trois types d'attaques, dont deux par collusion, portant sur les protocoles de personnalisation de contenus. Ces protocoles doivent résister à celles-ci, indépendamment du fait que ces attaques soient malveillantes ou accidentelles (voir définition 1.1 et 1.2). Les différents types d'attaques sont des attaques par insertion, des attaques sur le code et des attaques par fusion. Dans chaque cas, le but de l'adversaire est de détruire l'empreinte ou de suffisamment la modifier pour qu'il ne puisse être identifié en cas de diffusion illégale d'œuvre (Figure 1.6).

Définition 1.1. Attaque malveillante: *Une* attaque malveillante *est une attaque volontaire d'une personne sur un schéma défini. Cette attaque a pour but de supprimer, modifier ou altérer l'empreinte insérée dans le contenu pour qu'il puisse être redistribué par la personne sans que celle-ci ne soit inquiétée.*

Définition 1.2. Attaque non malveillante: *Une* attaque non malveillante *est une attaque non volontaire d'un utilisateur qui veut modifier le contenu (par exemple par un redimensionnement ou une compression).*

Pour les différentes méthodes d'attaques présentées ci-dessous, nous allons utiliser trois utilisateurs faisant partie de la collusion.

Alice	1	0	0	1	1	0	1
Bob	0	0	1	1	0	1	1
Charlie	1	0	1	0	0	0	1

Tableau 1.1.: Trois membres de la collusion avec leurs empreintes respectives.

Pour chacune des méthodes d'attaques présentées ci-dessous, nous décrivons une attaque spécifique et le résultat de celle-ci. Nous considérons que les adversaires ne savent pas quelles sont les empreintes insérées dans leurs contenus, car ils ne disposent pas de la clef permettant l'extraction de l'empreinte. Bien qu'ils ne les connaissent pas, ils peuvent comparer leurs copies et savoir où se situent les différences ce qui leur permet de mener les attaques décrites ci-après. De plus, on considère que chaque copie est découpée en m blocs et que chaque bloc contient un bit de l'empreinte des acheteurs. Ainsi lorsque deux acheteurs comparent leurs copies durant une collusion, il est possible qu'ils ne voient pas où le bit de l'empreinte se trouve dans la copie, car ce bit peut être identique dans les deux copies. Pour les attaques par fusion et celles sur le code, l'≪ *hypothèse de marquage* ≫ ou *Marking Assumption* (en Anglais)

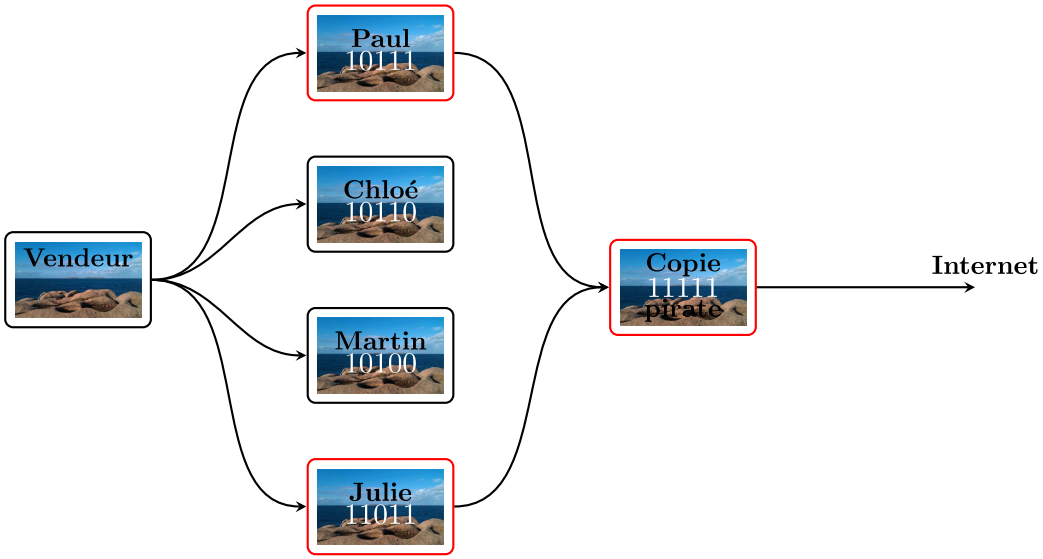

FIGURE 1.6.: Diffusion illégale d'une oeuvre. L'image est personnalisée et distribuée à deux personnes qui l'ont acquise légalement. La personnalisation est différente pour chacune d'elles. Paul et Julie forment une collusion et redistribuent l'image obtenue sur Internet. N'importe quel utilisateur peut donc la récupérer. L'empreinte contenue dans la nouvelle image est un « mélange » des empreintes des images ayant servi à faire la collusion.

s'applique. Cette dernière définit qu'en cas de collusion, les adversaires ne peuvent mettre un symbole (bit) dans la copie pirate si celui-ci ne fait pas partie d'une des copies de la collusion. L'intuition derrière cette hypothèse est que si toutes les copies des membres de la collusion comportent toutes la même valeur de bit pour un bloc particulier, il n'existe aucun moyen pour eux d'obtenir le bit inverse en appliquant une fonction de mélange sur ces blocs. Cependant, la *Marking Assumption* n'est pas valable pour les attaques sur le signal (section 1.4.3). La définition 1.3 formalise cette hypothèse. L'hypothèse de marquage fût pour la première fois définie par Boneh et Shaw, dans [22].

Définition 1.3. Hypothèse de marquage: *Considérons un ensemble d'acheteurs* $\{B_1, \ldots, B_c\}$ *réalisant une collusion de taille* c *sur un contenu* I_t *découpé en blocs d'indice* i *marqués* $\mathsf{fb}_B^{i,b}$ *où l'empreinte est tirée de l'alphabet* $\chi = \{0, 1\}$ *de taille* $|\chi| = 2$ *et* b *étant le bit de l'empreinte tirée à l'indice* i*. Pour tout* B *faisant partie de la collusion et ayant le même bloc personnalisé avec le bit* b *pour une position* i *(par exemple,* $\mathsf{fb}_{B_1}^{i,1} = \mathsf{fb}_{B_2}^{i,1} = \ldots = \mathsf{fb}_{B_c}^{i,1}$*) le bloc* $\mathsf{fb}_B^{i,b}$ *sera dans la copie pirate* FI*.*

Il existe trois types d'attaques décrites ci-après. Celles-ci sont classées par le moyen de les contrer. Pour contrer ces attaques il faut soit utiliser un tatouage robuste (par exemple [134], soit un code anti-collusion soit les deux.

1.4.1. Attaque sur la robustesse

Ce type d'attaque vise à effacer ou suffisamment modifier l'empreinte présente dans la copie de l'utilisateur en exploitant une faille dans la robustesse de la technique de tatouage. Si cette attaque réussit, la compromission de la méthode de tatouage est totale. Ainsi, une fois la personnalisation détruite, il sera impossible de tracer le traître lorsqu'il redistribuera illégalement le contenu acquis. Ce type d'attaque peut utiliser les traitements suivants : compression (par exemple, conversion du format RAW en JPEG pour que le poids de l'image soit moindre), agrandissement, impression puis numérisation de l'image, filmer le film diffusé au cinéma, ... Ces attaques sont dénommées *attaques sur la robustesse*, et ont la propriété d'attaquer directement le signal porteur. Elles peuvent être réalisées par un individu seul (autrement dit elles ne requièrent pas une collusion de deux acheteurs ou plus).

Une autre manière de faire, étudiée, par Cayre, Fontaine et Furon dans [28, 29, 30], est de retrouver la clef utilisée par la méthode de tatouage dans le but d'effacer proprement l'empreinte insérée. Pour ce faire, il faut que l'adversaire soit en possession de plusieurs documents ayant été tatoués en utilisant la même clef et qu'il connaisse l'algorithme de tatouage (principe de Kerchkoff). Ce type d'attaque est contrecarré par la technique de tatouage qui doit être aussi robuste que possible.

Un résultat possible, pour l'utilisateur Alice avec une attaque par compression, est :

Alice :	1	0	0	1	1	0	1
f_* :	?	0	?	?	1	0	?

Tableau 1.2.: Exemple d'attaque sur la robustesse. Les « ? » signifient que lors de la compression, des fautes ont été insérées à la place des bits de l'empreinte d'Alice.

1.4.2. Attaques par collusion

Ce type d'attaque vise à forger une copie pirate contenant une empreinte qui est composée de morceaux des empreintes des autres utilisateurs. Par exemple, les membres de la collusion vont comparer leurs copies pour distinguer où se trouvent les bits de l'empreinte et forger la copie pirate en conséquence. Ils peuvent employer différentes techniques pour choisir quels sont les blocs qui seront dans la copie sans forcément connaître les bits insérés. Par exemple, ils peuvent choisir de mettre le bit qui possède la plus grande occurrence pour une position donnée (attaque par *majorité*) ou encore celui qui a l'occurrence la plus faible (attaque par *minorité*) ou encore un 1 s'il y a au moins un 1 dans une copie d'un membre de la collusion pour une position particulière voir aussi de le faire de manière aléatoire. Pour une image, cela revient à découper l'image en blocs qui contiennent pour chaque bloc un ou plusieurs bits de l'empreinte et à choisir quels seront les blocs qui feront partie de la copie pirate. Les attaques de type majorité, minorité sont des attaques d'échange de blocs. Pour contrer les attaques par collusion, il est nécessaire d'employer une empreinte ayant une structure spécifique. L'empreinte insérée sera en fait un mot de code provenant d'un code anti-collusion tels

que les codes concaténés de Boneh et Shaw [22] ou ceux probabilistes de Tardos [125]. De plus, il faut aussi que le tatouage soit robuste.

Nous montrons un exemple de résultat d'attaques par collusion entre Alice, Bob et Charlie, dans le tableau 1.3.

Alice	1	0	0	1	1	0	1
Bob	0	0	1	1	0	1	1
Charlie	1	0	1	0	0	0	1
f_{1*}	1	0	1	1	0	0	1
f_{2*}	0	0	0	0	1	1	1
f_{3*}	1	0	1	0	0	1	1

Tableau 1.3.: Exemple de collusion entre trois acheteurs. f_{1*} est l'empreinte dans la fausse copie lors de la collusion avec la stratégie *majorité*, f_{2*} pour la stratégie *minorité*, f_{3*} pour la stratégie *aléatoire*. Dans l'exemple donné, aux positions 2 et 7 il y a le même bit dans toutes les empreintes. $\ll 0 \gg$ dans le premier cas et $\ll 1 \gg$ dans le second, nous les retrouvons donc dans la copie pirate à cause de l'hypothèse de marquage.

1.4.3. Attaques par fusion

Le but des attaques par fusion est de combiner, au niveau du signal, différentes copies pour obtenir une copie pirate. La fusion entraîne des pertes et des effacements de symboles dans l'empreinte résultante. Si nous reprenons notre exemple, Alice, Bob et Charlie fusionnent leurs copies. La copie pirate pourrait contenir une empreinte f_* comme celle présente dans le tableau 1.4.

Alice	1	0	0	1	1	0	1
Bob	0	0	1	1	0	1	1
Charlie	1	0	1	0	0	0	1
f_* :	1	0	1	?	0	?	0

Tableau 1.4.: Exemple d'attaque par fusion

Un exemple type d'attaque par fusion est la moyenne au niveau du signal. Il est important de noter que la moyenne des pixels n'équivaut pas à la moyenne des bits des empreintes. Seul un tatouage robuste permet de résister à ces attaques.

1.5. Codes anti-collusion

Dans cette section, nous allons introduire les codes anti-collusion. Nous présentons certaines solutions de la littérature tels que les codes de Boneh et Shaw [22, 23] et ceux de Tardos [125] avec les améliorations apportées sur ces derniers par Škorić, Katzenbeisser et Celik [131, 128].

1.5.1. Objectif et sécurité des codes anti-collusion

Objectif. Les codes anti-collusion servent, comme leur nom l'indique, à contrer une collusion d'utilisateurs malveillants. Plus précisément, leur objectif principal est de pouvoir retrouver avec une probabilité importante au moins une personne ayant participé à la collusion, pour pouvoir l'accuser de redistribution illégale de contenu. Nous nous intéresserons tout particulièrement, dans un premier temps, aux codes anti-collusion présentés par Boneh et Shaw en 1995 et 1998 [22, 23] ainsi que la solution proposée par Tardos [125, 126] en 2003. Dans un deuxième temps, nous verrons rapidement les améliorations apportées pas Škorić *et co-auteurs* [128].

Propriétés de sécurité. Les différentes propriétés de sécurité pour la personnalisation de contenus sont listées ci-dessous :

- *Frameproof :* la collusion de plusieurs adversaires ne doit pas pouvoir produire le mot de code d'un utilisateur innocent.
- *Secure-frameproof :* deux collusions sur le même contenu, réalisées par deux groupes distincts d'adversaires, ne peuvent pas fournir une copie pirate contenant le même mot. On dit qu'un mot de code est c-*secure* lorsque deux collusions distinctes de taille au plus c sur le même contenu ne peuvent pas générer une copie pirate contenant le même mot.
- *Identification parent property (ou Identification de parent) :* le mot produit par une collusion peut être tracé jusqu'à l'identification d'un membre de la collusion (c'est-à-dire, un parent du mot extrait de la copie pirate).
- *Traçabilité :* un coupable est identifié à chaque décodage d'un mot extrait de la copie pirate.

Il faut noter que $frameproof \Rightarrow secure - frameproof \Rightarrow identification\ parent\ property \Rightarrow traceability$, avec $\Rightarrow$ désignant l'implication. De plus, les propriétés *traçabilité* et *identification de parent* assurent une traçabilité forte qui implique l'utilisation de mots de code long. Au contraire, les propriétés *frameproof* et *secure-frameproof* s'accommodent d'une traçabilité faible avec une probabilité d'accuser à tort un innocent $\mathbb{P}(accusation\ a\ tort) < \epsilon_1$, par exemple, ceux de Tardos.

Certains codes anti-collusion de la littérature s'appuient sur des codes correcteurs d'erreur, comme ceux de Boneh et Shaw [22, 23] présentés ci-dessous. Cependant, ce n'est pas toujours le cas, comme nous le verrons plus loin avec les codes de Tardos [125]. Il existe d'autres codes anti-collusion dans la littérature, mais nous présentons dans la suite ceux ayant eu le plus fort impact et ceux utilisés dans les protocoles que nous décrivons dans les sections 1.6 et 2.4.

1.5.2. Codes de Boneh et Shaw

Boneh et Shaw ont proposé, dans [22, 23] respectivement en 1995 et 1998, les premiers codes anti-collusions. Ce sont les premiers à définir les propriétés *frameproof* et *secure-frameproof* ainsi qu'à proposer une construction respectant celles-ci. Ces codes anti-

	$\overbrace{\text{Red.}}$	
B_1	11	1111
B_2	00	1111
B_3	00	0011
B_4	00	0000

B_1	111111
B_2	110101
B_3	100001
B_4	000000

Tableau 1.5.: Exemple pour f_1 de la solution proposée par Boneh et Shaw [22, 23] avant permutation à gauche et après permutation à droite. (Red. signifie ici redondance)

collusions sont dits concaténés, car ils utilisent un code interne ($f_1^{m_1}$ avec f_1 un mot de code de taille m_1) et un code externe ($f_2^{m_2}$ avec f_2 un mot de code de taille m_2). La taille du code inséré est donc de taille $m = m_1 + m_2$ et $f = f_1 + f_2$. f_1 est un code *secure-frameproof* et est c-*secure* avec une probabilité de non-accusation bornée par ϵ pour une taille de collusion maximum c tandis que f_2 est un code correcteur d'erreurs[8]. L'utilisation de code concaténé permet de réduire la taille du code qui est linéaire en fonction du nombre de membres de la collusion. Prenons l'exemple d'un code interne dans la table 1.5 et un nombre d'utilisateurs $n = 4$ avec une redondance d'information (permettant l'utilisation de code correcteurs d'erreurs) égale à 2 (partie gauche de 1.5). Une permutation au niveau des colonnes est ensuite effectuée (table 1.5 à droite) et celle-ci est gardée secrète. Chaque utilisateur a donc un nombre de 0 et de 1 différents. Ce code (f_1) est le code interne. Lors de la phase de décodage, l'utilisateur accusé sera celui ayant le code le plus proche de celui présent dans la copie pirate. La distance est calculée, par exemple, à l'aide de la distance de Hamming entre le mot de code de chaque utilisateur et celui repéré dans la copie pirate. Boneh et Shaw ont aussi introduit dans cet article fondateur la *marking assumption* (Définition 1.3).

1.5.3. Codes de Tardos

En 2003, Peikert, Shelat et Smith [101] ainsi que Tardos [125] ont proposé une borne, pour la longueur d'un mot de code à traçabilité faible, inférieure à celle existante à ce moment-là. Contrairement à Peikert *et co-auteurs* Tardos a proposé une construction de code anti-collusion binaire atteignant cette borne. Ces codes binaires sont probabilistes [125]. C'est-à-dire que, pour les mêmes paramètres en entrée, les mots de codes générés seront différents. Les codes de Tardos sont définis par les paramètres suivants : la taille de l'alphabet $|\chi| = 2$ avec $\chi = \{0, 1\}$, le nombre maximum n d'acheteurs, la taille maximum c d'une collusion d'acheteurs malveillants tolérés par les codes et la probabilité $\delta << 1$ (dans la littérature notée ϵ_1) d'accuser un acheteur innocent. La structure de ce code est si efficace que le risque réel pour un acheteur donné de se faire accuser à tort est plus faible que la limite théorique [31], conduisant à une très

8. Un mot de code provenant d'un code correcteur d'erreurs permet de corriger le mot de code en cas, par exemple, d'effacement. Pour plus d'informations sur les codes correcteurs d'erreurs, il existe le livre [86] proposé par MacWilliams et Sloane en 1977.

faible probabilité globale de fausse alarme [9] même quand nous choisissons $\delta = 0.001$. La longueur m du code est définie par $m = Ac^2\lceil\log(1/\delta)\rceil$ et le *seuil d'accusation* est de la forme $Z = Bc\lceil\log(1/\delta)\rceil$. Tardos a fixé les paramètres A et B à $A = 100$ et $B = 20$ sans expliquer comment ces paramètres ont été trouvés et s'ils sont optimaux ou non.

Les codes de Tardos comportent trois phases : *Initialisation*, *Construction* et *Accusation*.

1. *Initialisation :* Lors de la phase d'initialisation, le serveur génère le vecteur $\mathbf{p}$ avec m probabilités p_i tirées de manière indépendante et identiquement distribuées (avec $p_i \in [t, 1-t]$) avec $t = 1/300c$ et $p_i = sin^2 r_i$ en tirant de manière aléatoire et uniforme $r_i = [t', \frac{\pi}{2} - t']$ avec $0 < t' < \frac{\pi}{4}$ et $sin^2 t' = t$. La fonction de distribution (décrite originellement dans [131]) des p_i est :

$$f(p) = \frac{1}{\pi - 4t'} \frac{1}{\sqrt{p(1-p)}} \tag{1.1}$$

2. *Construction :* Le vendeur va générer une matrice X de taille $n \times m$. Chaque $f_{j,i}$ est tiré dans l'alphabet binaire $\chi = \{0, 1\}$ indépendamment. Chaque bit est tiré en respectant la loi de Bernouilli ($\mathbb{P}(f_{j,i} = 1) = p_i$). L'empreinte f_j correspond à un acheteur et est insérée dans le contenu numérique.
3. *Accusation :* Cette phase intervient lorsque le vendeur a trouvé une copie forgée d'un contenu. La fonction d'accusation g est utilisée pour calculer un score d'accusation, pour chaque acheteur B_j ayant acheté le contenu, noté S_j. Pour cela, le vendeur extrait l'empreinte f_* trouvée dans la copie pirate. Ensuite, il applique les formules 1.4 et 1.2 ou 1.3 entre f_* et toutes les empreintes f_j (contenues dans X) pour calculer les scores d'accusation (formule 1.4). À ce moment-là, seuls les acheteurs dont le score S_j est supérieur au seuil Z sont accusés. Dans ce cas, le nombre d'utilisateurs est borné par la taille maximum de la collusion c. Cependant, il serait aussi possible d'accuser de deux autres manières :
   — L'utilisateur B_j ayant le score le plus important est accusé. Il faut pour cela calculer les scores pour tous les acheteurs.
   — Dès que l'on trouve $S_j > Z$ pour un acheteur B_j alors, ce dernier est accusé. Il n'est pas nécessaire, dans ce cas, de calculer les scores d'accusation pour tous les utilisateurs.
   — L'utilisateur B_j ayant le score le plus important et étant supérieur à Z.
   Plus le score est important plus la probabilité que l'accusé soit coupable est forte. Ceci est dû au calcul du score qui est pondéré par les p_i. Le calcul des scores s'effectue sur les bits à 1 de l'empreinte f_* extraite de la copie pirate.

Furon, Guyader et Cérou ont montré dans [56] que ces fonctions d'accusation étaient optimales lorsque la stratégie des adversaires n'était pas connue, mais qu'elles pouvaient être améliorées dans le cas contraire. Charpentier, Xie, Fontaine et Furon dans [32] ont montré que lorsque la stratégie des adversaires était connue, alors il était possible de trouver des fonctions d'accusation optimisées qui réduisent la longueur du code.

9. Nous entendons par « fausse alarme » la probabilité qu'un acheteur innocent soit faussement accusé.

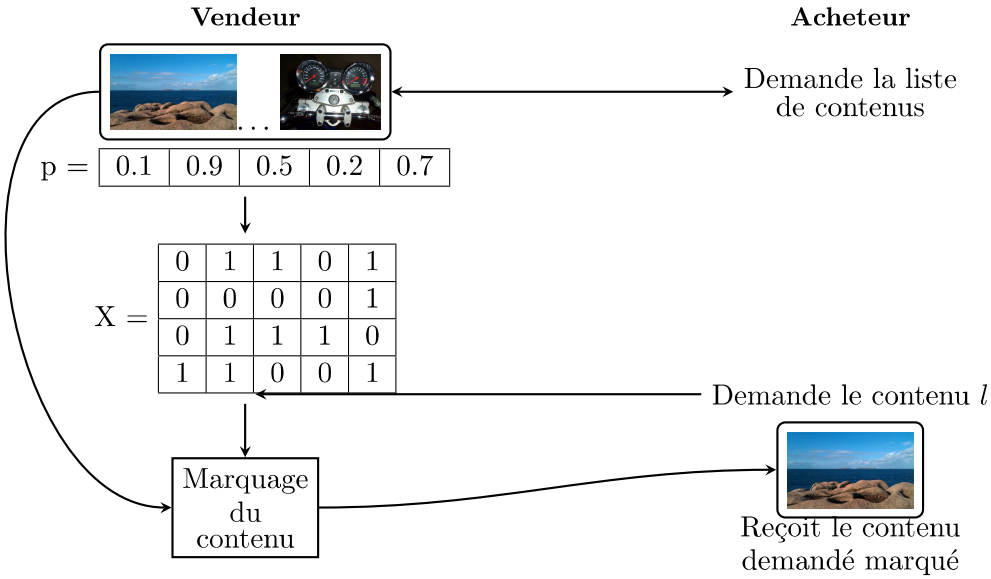

FIGURE 1.7.: Schéma illustrant l'utilisation des codes de Tardos.

Voir la figure 1.7 montrant l'utilisation des codes de Tardos dans un schéma symétrique.

$$g(1,1,p_i) = \sqrt{(1-p_i)/p_i} \tag{1.2}$$

$$g(1,0,p_i) = -\sqrt{p_i/(1-p_i)} \tag{1.3}$$

$$S = \sum_{i=1}^{m} g(f_{i*}, \mathsf{X}_{\mathsf{ji}}, p_i) >^{?} Z \tag{1.4}$$

Exemple 1. *Nous présentons ici un exemple des codes de Tardos. Dans cet exemple, le nombre d'utilisateurs est* $n = 5$*, la taille de la collusion est* $c = 3$ *et la taille des mots de code* $m = 10$*. Les équations utilisées pour calculer les scores sont les équations 1.2, 1.3 et 1.4. Il n'y a donc que les bits à 1 qui sont utilisés. Considérons le vecteur* $\mathbf{p}$ *(Tableau 1.6) et la matrice* X *(Tableau 1.7).*

0.7705	0.6274	0.2884	0.3838	0.1480	0.6878	0.8536	0.1249	0.4305	0.5305

Tableau 1.6.: Vecteur $\mathbf{p}$: Le vecteur $\mathbf{p}$ est composé des probabilités p_i avec $i \in [0..9]$

De plus, considérons une collusion entre les utilisateurs 0, 2 *et* 4 *possédant les empreintes* 0 1 0 0 0 1 1 1 0 0 , 0 1 0 0 0 1 1 0 0 1 *et* 1 1 1 0 0 0 0 0 1 0 *respectivement. S'ils utilisent une stratégie de type majorité lors de la collusion, l'empreinte* f_* *extraite*

0	1	0	0	0	1	1	1	0	0
1	1	0	0	1	1	0	1	0	0
0	0	0	0	0	1	1	0	0	1
1	1	1	0	0	0	1	0	0	1
1	1	1	0	0	0	0	0	1	0

Tableau 1.7.: Matrice composée des empreintes des utilisateurs $j \in [0..4]$ (une ligne correspond à un utilisateur). Chaque colonne correspond aux bits ayant comme indice $i \in [0..9]$ tiré aléatoirement et indépendamment en accord avec les p_i.

de la copie pirate sera : $Y = 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0$. *Le score calculé n'est pas linéaire, car il dépend de la valeur des* p_i. *En effet, si l'on prend* $p_8 = 0.1249$, *il y a une plus grande probabilité que les bits tirés à cet indice soient des* 0. *Donc si un* f_j *possède un 1 et qu'il y a un* 1 *dans* f_* *alors, le score pour ce bit sera élevé et par conséquent le score général augmentera plus vite que s'il y avait un* 0 *dans* f_j. *Les scores pour les utilisateurs de notre exemple sont :*

1.8585	-0.9702	-0.2097	-0.2995	-3.1225

Tableau 1.8.: Scores S_j (formule 1.4) des différents utilisateurs B_j pour leurs empreintes f_j calculées d'après les équations 1.2 et 1.3.

Comme on peut le voir, l'utilisateur 0 *possède le score le plus élevé et il fait bien partie de la collusion. Le deuxième score le plus élevé est celui de l'utilisateur* 2 *qui fait aussi partie de la collusion possède un score négatif. Comme on peut le voir, le dernier membre de la collusion (utilisateur* 4*) possède le score le moins élevé. Les utilisateurs 2 et 4 ne seront pas accusés tout comme l'utilisateur* 2, *car son score est négatif. Nous retrouvons bien au moins un membre de la collusion. Ici, nous accuserions plus particulièrement l'utilisateur* 0 *qui fait bien partie de la collusion.*

1.5.4. Variantes des codes de Tardos

Commentaires sur les codes anti-collusion. Les codes anti-collusions ont été largement étudiés dans la littérature et plus spécifiquement ceux de Tardos. Toutes ces études n'améliorent pas les codes de Tardos de manière globale, mais plutôt une partie spécifique. Par exemple, certains chercheurs ont essayé d'améliorer les codes de Tardos, par exemple, en regardant comment estimer quelle attaque a été faite par les adversaires pour agir en conséquence au moment de l'accusation [55, 34], de réduire la longueur des mots de codes [95] ou encore de rendre les codes anti-collusion q-aires [128]. Cela signifie que l'alphabet n'est plus de taille $|\chi| = 2$ avec $\chi = \{0, 1\}$, mais $|\chi| = q$ avec $\chi = \{0, ..., q-1\}$. De plus, Laarhoven et de Weger ont proposé, en 2014 [77], un schéma basé sur les codes anti-collusion de Tardos. Plus précisément, leurs solutions s'appuient sur l'amélioration des codes de Tardos proposée par Škorić *et co-auteurs* en 2008, pour la fonction d'accusation qui est symétrique, combiné avec l'analyse établie par Blayer et Tassa [19] la même année. Škorić, Vladimirova, Celik et Talstra [131],

Škorić, Katzenbeisser et Celik [128] pour les codes de Tardos q-aires, ou Skoric *et co-auteurs* dans [131, 78, 119, 130, 97, 129]. Furon et Meerwald [89] ont proposé une approche différente au moment de l'accusation. Desoubeaux, Herzet, Puech et Le Guelvouit, ont aussi contribué à l'amélioration de l'accusation des codes de Tardos dans [46]. Chaque amélioration n'améliore qu'une partie des codes de Tardos (longueur du mot de code, accusation, etc). Nous ne rentrerons pas dans le détail de ces améliorations dans ce mémoire excepté les améliorations fournies par Škorić, Katzenbeisser et Celik dans [131]. En effet, dans nos contributions nous pouvons utiliser n'importe quelle solution des codes de Tardos améliorés (moyennant potentiellement un ajustement au niveau de la phase de préparation des contenus et de l'accusation) suivant le but recherché.

Škorić, Katzenbeisser et Celik. Dans la littérature, on peut trouver beaucoup de tentatives pour réduire les constantes A et B. En 2008, Škorić, Vladimirova, Celik et Talstra, dans [131], améliorent cette dernière en rendant l'accusation symétrique. Cela a pour conséquence de réduire la taille des mots de codes. Une accusation symétrique consiste à utiliser tout le mot de code (tous les bits 0 et 1). En effet, Tardos n'employait que les bits à 1 dans l'empreinte f_*. Les fonctions d'accusation symétrique sont données dans les formules 1.9 et 1.8. L'accusation est efficace, peu importe la stratégie de la collusion des acheteurs. De plus, le score pour chaque bit est pondéré par les probabilités p_i qui ont une distribution symétrique sur 0.5 dont la majorité est proche de 0 et de 1 (Distribution des probabilités p décrite par Škorić *et co-auteurs* formule 1.7). Cela implique que si un bit d'un utilisateur pour une position i est le même que dans la copie pirate, la probabilité que celui-ci fasse partie de la collusion est importante, ce qui a pour effet d'augmenter son score. À l'inverse, si les bits sont différents la probabilité que l'utilisateur fasse partie de la collusion est faible. Cependant, il est à noter que le fait que les bits soient différents ne discrimine pas un utilisateur (et réciproquement).

Pour finir, l'amélioration de la phase d'accusation conduit à la réduction de la taille des mots de code. Ceci est important car les codes de Tardos sont plus performants et plus utilisables en pratique (*eq.* 1.5 et 1.6 avec $A = 2\pi^2$ et $B = 2\pi$).

$$m = 2\pi^2 c^2 [\ln(1/\varepsilon_1)] \tag{1.5}$$

$$Z = 2\pi c \lceil \ln(1/\delta) \rceil \tag{1.6}$$

$$f(p) = \frac{1}{2arcsin(1-2t)} \frac{1}{\pi\sqrt{p(1-p)}} \tag{1.7}$$

$$g(1,1,p_i) = g(0,0,1-p_i) = \sqrt{(1-p_i)/p_i} \tag{1.8}$$

$$g(1,0,p_i) = g(0,1,1-p_i) = -\sqrt{p_i/(1-p_i)} \tag{1.9}$$

Exemple 2. *Nous présentons un exemple des codes de Tardos avec les améliorations de*

Škorić et co-auteurs. *Nous reprenons la matrice* X *(Tableau 1.7) de l'exemple précédent ainsi que tous les autres paramètres (m, c,* **p***, X, les utilisateurs et les membres de la collusion) et nous utilisons les équations 1.8 et 1.9. Les scores calculés avec ces nouvelles équations sont présentés dans le tableau 1.9. Comme on peut le voir, l'utilisateur*

9.3546	1.3316	8.3073	3.6321	0.7874

Tableau 1.9.: Scores S (*eq.* 1.4) des différents utilisateurs B_j pour leurs empreintes f_j calculées d'après les équations 1.8 et 1.9.

0 *possède le score le plus élevé et il fait bien partie de la collusion, ainsi que le deuxième score le plus élevé (utilisateur* 2*). Par contre, les utilisateurs* 1 *et* 3 *possèdent, respectivement, un score de* 1.3316 *et* 3.6321 *ce qui est supérieur au score du dernier membre de la collusion (utilisateur* 4*). Cela implique que si on veut accuser trois personnes, nous accuserions un acheteur innocent. Cependant, les codes de Tardos offrent comme garantie principale de retrouver au moins un membre de la collusion avec une forte probabilité. Ici, nous accuserions l'utilisateur* 0 *qui fait bien partie de la collusion dans le cas où nous calculons les scores de tous les utilisateurs.*

Comparaison des résultats. Comme on peut le constater, les résultats obtenus avec la méthode de Tardos originelle et les modifications apportées par Škorić *et co-auteurs* sont différents. En effet, les scores de la deuxième méthode sont plus élevés qu'avec la première. Cependant, dans ces exemples, si nous classons les utilisateurs par scores obtenus alors on s'aperçoit que les utilisateurs sont classés dans le même ordre (classement des utilisateurs : 0, 2, 3, 1, 4). Cela implique que dans les deux cas on retrouve bien l'un des adversaires, mais aussi que l'utilisateur 4 n'est pas inquiété, car son score est le plus bas bien que cet utilisateur fasse partie de la collusion. L'intérêt de la solution de Škorić *et co-auteurs* par rapport à celle de Tardos est le fait d'améliorer l'accusation, ce qui a pour conséquence de permettre l'utilisation de mots de code plus court.

1.6. Détails sur certains protocoles de personnalisation de contenus existants

1.6.1. Schémas symétriques contre protocoles asymétriques et protocoles acheteur-vendeur

Symétriques. Les protocoles symétriques n'existent pas vraiment. En effet, un protocole requiert des interactions entre différentes entités. Or, lorsque dans les propositions dites symétriques, les deux seules interactions sont : la demande du contenu de l'acheteur auprès du vendeur, et la réception du contenu envoyé par le vendeur à l'acheteur. En considérant cela, nous parlerons de schéma pour le cas symétrique et de protocole pour les cas asymétriques et anonymes.

Wagner, dans [133] propose un exemple de ce qui peut être fait pour personnaliser des données numériques (entier, flottant, etc). Ce qui implique que ses explications ne

prennent pas en compte les contenus étant des images (images, films, etc), mais plutôt les bases de données ou encore des tables mathématiques (par exemple, table de Néper). Sa proposition est simple : il suffit de modifier la valeur originale ($VOrg_j$) par une valeur dans l'intervalle $[VOrg_j - v_j, VOrg_j + v_j]$ avec $v_j > 0$ étant le delta d'une valeur j, delta étant le même pour tous les utilisateurs. La valeur v doit être suffisamment petite pour ne pas dégrader l'utilité des données. Le choix d'utiliser $Vorg_j$, $VOrg_j + v_j$ ou $VOrg_j - v_j$ est déterminé de manière aléatoire. Wagner est aussi le premier à considérer qu'un utilisateur ne doit pas modifier aveuglément les données sous peine de leur faire perdre toute utilité. L'avantage de cette solution est le fait que la personnalisation est rapide. Wagner évoque aussi dans son article, le problème de la personnalisation de contenus. En effet, Wagner conclut que l'empreinte doit être adaptée au support et que celle-ci doit être pseudo-aléatoire en prenant comme graine l'identité de l'utilisateur. Depuis Wagner, des schémas symétriques ont été proposés, comme [24] par Brassil, Low, Maxemchuk et O'Gorman qui personnalisent un contenu textuel en jouant sur l'espace entre les lignes et entre les mots.

Du fait de leur faiblesse, nous ne discuterons pas davantage des schémas symétriques pour nous concentrer dans la suite de ce mémoire sur les protocoles asymétriques et anonymes. Il est aussi important de noter que le mot symétrique n'a pas le même sens suivant le contexte. Pour le chiffrement, symétrique signifie que la même clef est utilisée pour chiffrer et déchiffrer. Un schéma symétrique (dans le cas de la personnalisation de contenus) signifie que le vendeur génère l'empreinte avant de l'insérer dans le contenu vendu et de l'envoyer à l'acheteur. La dernière interprétation possible (celle utilisée dans les codes anti-collusion) est utilisée pour indiquer que l'accusation est symétrique. Dans ce cadre, cela signifie que l'empreinte pirate complète est utilisée à ce moment du protocole (qui peut être asymétrique).

Asymétriques. Le premier protocole asymétrique de personnalisation de contenus fut présenté par Pfitzmann et Schunter [105] en 1996. Depuis ce jour, de nombreux autres ont vu le jour [107, 91, 92]. Nous ne détaillerons pas tous ceux existants pour nous concentrer sur les plus importants de la littérature.

Birgitt Pfitzmann et Matthias Schunter [105] sont partis d'un schéma symétrique (c'est-à-dire, ayant deux algorithmes `IBuy` et `Identify`) puis ils ont ajouté diverses propriétés, expliquées ci-après, pour rendre celui-ci asymétrique.

1. *Sans mémoire :* l'algorithme `IBuy` ne doit pas utiliser une empreinte en fonction de l'identifiant de l'utilisateur, mais une empreinte aléatoire. Cependant, ils conservent secrets les couples contenant les coordonnées des bits constituant les empreintes dans l'image.
2. *Espace d'identifiant large :* permet de ne pas avoir une même empreinte (identifiant) pour deux utilisateurs différents. Cet espace est noté par la suite Id_Space.

Avant de fournir une solution pratique, Pfitzmann *et co-auteurs* proposent une construction générale d'un protocole de personnalisation de contenus asymétrique respectant les restrictions précédentes et les propriétés de *Conformité*, de *Traçage de traîtres* et d'*Anti-framing*.

Leur construction générale est composée de quatre sous-protocoles :

1. `Setup` : dans ce protocole l'acheteur va créer une paire de clefs privée/publique ($\mathsf{sk_B}$,$\mathsf{pk_B}$) d'un schéma de signature.
2. `IBuy` : ce protocole permet de générer une empreinte personnalisée par utilisateur et de s'en servir pour personnaliser le contenu qui sera remis à l'acheteur. Il s'agit d'un protocole biparti entre l'acheteur et le vendeur. Cette partie est composée des sous-protocoles suivants :
   — `Fing` permettant de personnaliser le contenu vendu.
   — `Sign` permettant à l'acheteur de signer le contrat de la vente.
   — `Verify` permettant de vérifier la signature de l'acheteur.
   Dans ce protocole, le contenu est une image. Tout d'abord, l'acheteur B choisit une identité id aléatoirement dans l'espace possible des identités avant de réaliser un protocole biparti entre le vendeur et l'acheteur pour un contenu I donné. Le vendeur fournit I, $record\_list_I$, $\mathsf{pk_B}$ et $text_M$ qui correspondent respectivement au contenu original, à la liste d'enregistrement des ventes de ce contenu et au contrat de vente. L'acheteur fournit id, $\mathsf{sk_B}$ et $text_B$. Le protocole biparti réalise l'insertion de id dans I. Il vérifie que $text_M = text_B$ avant d'appliquer une fonction de hachage sur id ($hash(id)$), qu'il concatène avec $text$ ($msg_B := (hash(id), text)$) et signe avec $\mathsf{sk_B}$ (σ_I^B). Ensuite, il vérifie σ_I^B avec $\mathsf{pk_B}$ (fourni par le vendeur) et donne $record_M = (\mathsf{pk_B}, text, hash(id), \sigma_I^B)$ au vendeur et le contenu personnalisé à l'acheteur.
3. `Identify` : Cet algorithme est utilisé par le vendeur pour identifier l'acheteur du contenu qui a été redistribué. Pour cela, le vendeur fournit $\mathsf{FI}^*_{\mathsf{B}^*,\mathsf{I}}$, I et $record\_list_I$, $\mathsf{pk_B}$. Si la sortie est id^* alors il continue. Si la sortie est $\perp$ alors il arrête le protocole. Pour le premier cas, il calcule $hash(id^*)$ et vérifie si $hash(id) = hash(id^*)$ est présent dans sa liste d'enregistrement de vente. Si la sortie de cet algorithme identifie un acheteur, la sortie est $\mathsf{pk_B}$, $text$ et une preuve $\pi = (id^*, \sigma_I^B)$. Dans ce cas, le vendeur utilise le protocole d'accusation `Accuse`.
4. `Accuse` : Il s'agit d'un protocole entre deux ou trois participants : le vendeur M, le juge J et potentiellement l'acheteur B. Ce protocole permet d'accuser un acheteur particulier qui a été préalablement identifié avec l'algorithme `Identify`. Pour cela, le vendeur fournit une preuve de la culpabilité d'un acheteur en entrée du protocole. Le juge J vérifie que la preuve est bien constituée de id^*,σ_I^B et ensuite il vérifie la signature σ_I^B avec $\mathsf{pk_B}$ sur $msg_B := (hash(id), text)$.

Dans cette solution, les auteurs ne tiennent pas compte des collusions. En effet, en cas de collusion la condition $hash(id) = hash(id^*)$ ne retournera aucune identité. En se basant sur la solution précédente, les auteurs proposent une construction plus spécifique, basée sur *l'engagement de bits* (ou *Bit Commitment* en anglais, BC), détaillée ci-dessous :

1. `Setup` : dans ce protocole l'acheteur va créer une paire de clefs privée/publique ($\mathsf{sk_B}$,$\mathsf{pk_B}$) d'un schéma de signature.

2. `IBuy` : Le vendeur commence par regarder où il peut insérer l'empreinte dans le contenu I. Il ne fait cela que pour le premier acheteur, car il stocke les positions du contenu pouvant accueillir l'empreinte dans pos. Ensuite, il initialise un compteur $counter$ à 0 qui est de taille len_1. Après cela, pour chaque acheteur le protocole se déroule ainsi :
   — Le vendeur envoie la valeur courante de $counter$ puis l'incrémente.
   — L'acheteur choisit une valeur aléatoire $id\_proof \in \{0,1\}^{len_2}$ et obtient une identité aléatoire $id := (counter, id\_proof) \in Id\_Space$ qui est l'empreinte insérée dans le contenu I. L'espace d'identifiants Id_Space résulte d'un algorithme déterministe. Ensuite, id est encodé $code(id)$ puis mis en gage $com := BC(code(id))$. Pour finir, l'acheteur signe le message avec $\mathsf{sk_B}$. Le message est constitué de com (un engagement de bits homomorphe) et $text$ ($msg_B := (com, text)$). Ensuite, l'acheteur envoie σ^B_{msg} et $msg_B := (com, text)$ au vendeur.
   — L'acheteur prouve avec une preuve à divulgation nulle de connaissance qu'il connaît une valeur id dont la première partie est $counter$ et que com est bien un engagement sur $code(\mathsf{id})$.
   — Le vendeur vérifie la signature puis encode le contenu numérique en $code(I)$ et multiplie $com := BC(code(id))$ et $code(I)$ aux positions identifiées précédemment. Il envoie ensuite $Com(\mathsf{FI})$ à l'acheteur qui le déchiffre et qui obtient FI. Il conserve aussi $text$, $counter$, id et les informations permettant d'ouvrir l'engagement. Le vendeur, quant à lui, conserve $record_M = (counter, \mathsf{pk_B}, text, msg, \sigma^B_{msg})$ et le stocke dans $record\_list$. Le vendeur n'a aucune information sur l'empreinte insérée dans le contenu à ce moment du protocole.
3. `Identify` : Cet algorithme est utilisé par le vendeur pour identifier l'acheteur du contenu qui a été redistribué. `Identify` prend en entrée la copie pirate $\mathsf{FI}^*_{\mathsf{B}^*,\mathsf{I}}$, le contenu original I et pos. S'il obtient en sortie $id := (counter, id\_proof)$ avec $counter$ présent dans $record\_list$ alors il récupère $record_M = (counter, \mathsf{pk_B}, text, msg, \sigma^B_{msg})$ et obtient en sortie $\mathsf{pk_B}$, $text$ et $\pi = (id, msg, \sigma^B_{msg})$ où π est une preuve que σ^B_{msg} est valide pour le message msg avec la clef $\mathsf{pk_B}$ et que la deuxième partie du message est bien $text$. Le vendeur envoie toutes ces informations au juge J.
4. `Accuse` : Ce sous-protocole permet à un juge de déterminer la pertinence des informations reçues par le vendeur. Tout d'abord, le juge J vérifie π. Si la vérification échoue, il avorte la procédure de vérification. Sinon, il vérifie que $code(id)$ est contenu dans com. Pour cela il demande à l'acheteur de fournir une preuve à divulgation nulle de connaissance qu'au moins un bit de com n'est pas le bit correspondant dans $code(id)$. Si l'acheteur réussit à prouver cela alors il est considéré comme innocent.

Comme on peut le voir, l'acheteur génère lui-même l'empreinte qui est insérée dans le contenu et le vendeur n'a aucune information sur celle-ci (excepté la valeur compteur). De plus, les auteurs considèrent leur protocole comme sûr si les

techniques cryptographiques (génération des clefs, signature de messages,...) le sont. Leur solution n'est pas vraiment résistante aux collusions, car l'empreinte extraite est $id := (counter, id\_proof)$. Or, en cas de collusion, *counter* pourrait être altéré et donc ne pointer vers aucun tuple $record_M = (counter, \mathsf{pk_B}, text, msg, \sigma^B_{msg})$ ou alors vers un tuple d'un autre acheteur de la base de stockage. Cependant, les auteurs énoncent rapidement comment faire pour intégrer les codes concaténés de Boneh et Shaw [22].

Dans [107], Pfitzmann et Waidner exposent plusieurs solutions plus ou moins viables en ce qui concerne le traçage de traîtres. Pour cela, ils s'appuient sur le schéma symétrique de Chor, Fiat et Naor [39] pour proposer une construction d'un protocole asymétrique. La résistance aux collusions de leur solution provient de [39]. L'idée de ce protocole est que l'acheteur choisit lui-même son mot de code, de taille m, aléatoirement tiré d'un alphabet de taille $|\chi|$ et résistant à une collusion d'au plus c membres. Le vendeur apprend une partie de celui-ci, qui est suffisante pour identifier des traîtres. Pour les paramètres $m = 64c(\epsilon + log_2(n))$, $N = \{1, \ldots, 48c\}$ et $|\chi| = \{1, \ldots, N\}$ avec ϵ la probabilité d'accuser un innocent, n le nombre maximum d'acheteurs pouvant acquérir le contenu et $|\chi|$ l'alphabet de taille N. Leur solution fonctionne de la manière suivante :

— `Setup` : L'acheteur génère une paire de clefs privée/publique ($\mathsf{sk_B}$,$\mathsf{pk_B}$) d'un schéma de signature et il distribue $\mathsf{pk_B}$, tandis que le vendeur choisit m ensembles S contenant chacun N clefs $\mathsf{KP_{N,m}}$ d'un schéma de chiffrement symétrique. On peut voir cela comme une matrice, de taille $m \times N$, contenant des clefs de chiffrement symétrique.

— `IBuy` : L'acheteur choisit un mot de code f_B de taille m sur $|\chi|$ puis engage celui-ci (com_B) en l'envoyant au vendeur. Ensuite, l'acheteur signe le message $msg_B := (text, com_B)$ avec $\mathsf{sk_B}$ ce qui donne σ^B_I où *text* une description expliquant sur quoi porte la signature. L'acheteur envoie σ^B_I et le vendeur vérifie la validité de cette signature avec $\mathsf{pk_B}$. Ensuite un protocole de calcul biparti est exécuté dans lequel les entrées sont : pour l'acheteur com_B et $open_B$ où $open_B$ correspond aux informations permettant d'ouvrir l'engagement, et pour le vendeur $m \times |\chi|$ clefs secrètes générées lors de la phase `Setup` ainsi qu'un ensemble aléatoire Set_B d'indices de taille $\frac{m}{2}$ et com_B. Ce protocole biparti vérifie que $open_B$ ouvre com_B pour révéler le mot de code ainsi que Set_B est bien de taille $\frac{m}{2}$. Set_B est utilisé pour sélectionner les bits du mot de code pour fournir Hw_B (c'est-à-dire, le *halfword*) au vendeur tandis que les m clefs correspondantes à f_B sont fournies à l'acheteur. Le vendeur enregistre $record_M = (id_B, text, com_B, \sigma^B_{I_B}, Set_B, \mathsf{Hw}_B)$ alors que l'acheteur reçoit $record_B = (text, f_B, open_B)$. À ce moment-là, l'acheteur n'a toujours pas obtenu le contenu.

— `IRecover` : Le vendeur envoie à tous les utilisateurs légitimes, les données chiffrées avec une clef de session SKey ainsi que cette dernière découpée en bloc BSKey où chaque bloc est chiffré. Le découpage est en fait un Xor de taille m. C'est-à-dire que le Xor entre m chaînes donne la clef de session. Ensuite, chaque chaîne (bloc) de la clef de session est chiffrée avec chacune des clefs des pots. En d'autres termes, chaque SKey est découpée en blocs BSKey qui sont chiffrés $\mathsf{Enc}_{\mathsf{KP_{N,m}}}(\mathsf{BSKey})$ avant d'être envoyés. Ensuite, chaque acheteur déchiffre avec les clefs $\mathsf{KP_{N,m}}$ reçues

précédemment et reconstitue la clef de session SKey ($\mathsf{SKey} = \mathsf{BSKey_1} \oplus \mathsf{BSKey_2} \oplus \cdots \oplus \mathsf{BSKey_m}$). Cette dernière leur permet d'accéder au contenu.

— `Identify` : Cette étape a lieu lorsque le vendeur trouve une copie forgée. Les auteurs supposent qu'au moins une clef de chaque ensemble est retrouvée avec leurs coordonnées. Ensuite, pour retrouver un membre de la collusion, ils recherchent au moins $\frac{m}{4c}$ symboles communs entre f_* et les paires (Set_B, Hw_B) de chaque acheteur. Si la condition est atteinte, alors le vendeur génère une preuve contenant $\pi = (msg_B, \sigma_I^B, f_*)$ de la culpabilité de l'acheteur.

— `Accuse` : Le vendeur envoie π au juge. Tout d'abord, le juge vérifie la signature avec $\mathsf{pk_B}$ puis le vendeur montre que f_* a au moins $\frac{m}{2} + \frac{m}{16c}$ symboles en commun avec com_B. Si ce n'est pas le cas, l'acheteur ouvre com_B.

Dans le même article, Pfitzmann et Waidner proposent une construction où le mot de code f provient des codes concaténés de Boneh et Shaw [22]. Considérant que m est la taille du mot de code externe $m = 64c(\epsilon + log_2(n))$ et l'alphabet est $|\chi| = \{1, \ldots, 48c\}$, les sous-protocoles sont les suivants :

— `Setup` : L'acheteur génère une paire de clefs privée/publique ($\mathsf{sk_B}$,$\mathsf{pk_B}$) d'un schéma de signature et il distribue $\mathsf{pk_B}$, tandis que le vendeur choisit les positions du contenu I qui recevront l'empreinte.

— `IBuy` : Cette étape s'effectue de la même manière que pour le protocole précédent à l'exception que l'empreinte insérée vient d'un code interne (f_1 de taille $\frac{m}{2}$) généré par le vendeur et d'un code externe (f_2 de taille m) généré par l'acheteur. Ce dernier est mis en correspondance du code interne et est généré par l'acheteur de manière aléatoire sur $|\chi|$. La dernière différence est que l'acheteur obtient une copie personnalisée du contenu au lieu de seulement des clefs. La phase `IRecover` n'est donc plus nécessaire.

— `Identify` : La première étape est l'identification du code interne f_1 avant de procéder comme dans la phase `Identify` du protocole précédent.

— `Accuse` : Cette dernière étape est effectuée comme `Accuse` ci-dessus.

En ce qui concerne le traçage de traîtres, cette deuxième solution est plus pertinente. Nous entendons par là que la probabilité de retrouver un traître est plus importante grâce à la structure du code inséré. De plus, les codes de Tardos ne peuvent être utilisés dans ces protocoles à cause de leur structure. Aussi, comme on peut le voir, Pfitzmann *et co-auteurs* ont surtout travaillé à l'intégration des codes concaténés de Boneh et Shaw [22] car ils étaient considérés comme étant les meilleurs à l'époque.

Protocoles de tatouage acheteur-vendeur (ou *Buyer-Seller watermarking protocol* en anglais). Memon et Wong ont proposé en 2001 dans [92] un protocole de tatouage acheteur-vendeur. Ce protocole est en fait un protocole de personnalisation de contenus (*fingerprinting protocol*). Dans leur protocole ils utilisent trois entités : l'acheteur, le vendeur et ce qu'ils appellent une autorité de certification de tatouage.

La première étape consiste au fait que l'acheteur demande une empreinte f, générée aléatoirement, valide en échange de son identité B et d'une clef publique $\mathsf{pk_B}$. L'empreinte est sous forme de vecteur V où chaque bit est chiffré de manière indépendante avec du chiffrement homomorphe puis signé σ_f, par l'autorité de certification de ta-

touage, avant d'être transmise à l'acheteur.

Ensuite, l'acheteur vérifie σ_f avant de transmettre V avec σ_f au vendeur. Ce dernier génère une empreinte personnalisée f^B pour la transaction avant d'être insérée dans le contenu. Ensuite, le vendeur permute les éléments de l'empreinte f avant de l'insérer dans l'image avant de l'envoyer à l'acheteur. L'acheteur reçoit donc une image chiffrée de manière homomorphe et contenant deux empreintes f et f_B. Cela implique que le vendeur n'a pas connaissance de l'image personnalisée. Dans ce protocole, l'acheteur connaît l'empreinte f, mais il ne connaît pas la permutation effectuée par le vendeur ni où celle-ci a été insérée. Le vendeur stocke l'identifiant de l'acheteur B, l'empreinte personnalisée générée f_B, l'empreinte générée par l'autorité f, V, σ_f et la permutation effectuée. L'acheteur recevant la copie personnalisée avec f et f_B déchiffre celle-ci avec $\mathsf{sk_B}$.

Lorsque le vendeur trouve une copie pirate, il extrait f_* de celle-ci et réalise une corrélation entre cette empreinte et f_B. Si l'acheteur suspect nie le fait d'avoir redistribué l'image, alors le vendeur envoie la permutation, f et σ_f au juge. Le juge commence par vérifier σ^B. Ensuite, il demande à l'acheteur de lui fournir le déchiffrement de f. Pour ce faire, le juge envoie f à l'acheteur qu'il peut déchiffrer avec $\mathsf{sk_B}$

Leur protocole possède différentes faiblesses. Premièrement, il n'est pas résistant aux collusions, car l'empreinte n'est pas conçue par un code anti-collusion. Deuxièmement, l'empreinte est générée par une tierce partie avant d'être fournie entièrement au vendeur et avant cela à l'acheteur. Cela implique que : 1) l'acheteur possède une information qu'il ne devrait pas connaître (l'empreinte), mais il ne connaît pas la permutation, 2) le vendeur peut tatouer un nouveau contenu avec f_B et f qui sont des empreintes valides. De plus, leur protocole permet au juge de chiffrer l'empreinte $f^{'}$ avec la clef publique de l'acheteur. Cela implique que l'algorithme de chiffrement est déterministe (c'est-à-dire, que chiffrer deux fois la même valeur donne en sortie le même chiffré). Le vendeur ne connaît normalement pas l'empreinte f. Mais l'algorithme de chiffrement étant déterministe, il peut tester des valeurs, les chiffrer et les comparer aux valeurs présentent dans f et ainsi connaître l'empreinte de l'acheteur.

Les protocoles asymétriques de personnalisation de contenus ont été beaucoup moins étudiés que les protocoles anonymes que nous aborderons dans la Section 2.4 après avoir parlé du respect de la vie privée dans le chapitre 2. De plus, les solutions présentées précédemment ne sont pas compatibles avec les codes de Tardos (excepté [92] du fait de la structure et de la nature probabiliste de ces derniers. Cela implique la construction spécifique de protocoles compatibles avec eux dont nous discutons dans la section 1.6.2.

1.6.2. Codes anti-collusion de Tardos dans les protocoles asymétriques de personnalisation de contenus pour tracer les traîtres

À notre connaissance, les codes de Tardos ont seulement été utilisés dans deux autres protocoles de personnalisation de contenus [33, 73], décrits ci-dessous. Le premier sera décrit plus en détail, car nous l'avons utilisé comme base de notre travail et qu'il nous paraît important de bien le comprendre avant d'aller plus loin.

Charpentier, Fontaine, Furon et Cox. Le protocole [33] utilise les codes de Tardos, mais l'empreinte est générée dans un protocole biparti entre le vendeur et l'acheteur au lieu d'être générée uniquement par le vendeur. [33] commence par une phase d'*initialisation* (Figure 1.8) dans laquelle, quand on donne en entrée les paramètres du code de Tardos, un algorithme de génération probabiliste retourne un vecteur $\mathbf{p}$ de taille m contenant des probabilités p_j aléatoires distribuées identiquement et indépendamment. Ces probabilités sont uniquement connues par le vendeur et gardées secrètes de toutes les autres entités. Ensuite, une *quantification* de ces valeurs est réalisée : $p_j = \frac{\mathsf{L}_j}{N}$ avec $\mathsf{L}_j \in [N-1]$, pour un entier N (typiquement $N = 20$). Le processus de quantification consiste à choisir un nombre L_j de bits égal à 1 et un nombre $N - \mathsf{L}_j$ de bits égal à 0 pour chacun des j. Les bits, notés $f_{I_t}^{i,j}$ pour $i \in \{1, \ldots, m\}$ et $j \in \{1, \ldots, N\}$, sont stockés dans une matrice de taille $N \times m$ nommée $\mathcal{F}_{I_t}$. Le vendeur génère et stocke $N \times m$ clefs de chiffrement symétrique noté $\mathsf{k}_{I_t}^{i,j}$ dans une matrice κ_{I_t} pour un contenu I_t. Ces bits $f_{I_t}^{i,j}$ et les clefs $\mathsf{k}_{I_t}^{i,j}$ sont ensuite utilisés pour générer une mémoire WORM [98] (*Write Once Read Many* ou *Écrire une fois Lire plusieurs fois*) $\tau_{\mathsf{I_t}}$ de taille $N \times m$ contenant les bits de personnalisation chiffrés $\mathsf{fb}_{I_t}^{i,j} = \mathtt{Enc}(\mathsf{k}_{I_t}^{i,j}, f_{I_t}^{i,j})$.

Durant la phase d'*achat* (Figure 1.9), l'acheteur et le vendeur génèrent une empreinte f_t personnalisée qui est ensuite insérée dans le contenu. Plus précisément, l'acheteur exécute un protocole de transfert équivoque (OT_1^N) avec le vendeur conduisant à la récupération, par l'acheteur, de m clefs $\mathbf{KI}_{B,I_t} = \{(i, \mathsf{k}_{I_t}^{i,j})_{j=1}^m\}$ (une par colonne m du WORM). Le vendeur fournit en entrée de cet OT, à chaque tour, une colonne de κ_{I_t}. Ces clefs permettent à l'acheteur de déchiffrer le contenu de m cellules du WORM autorisant la récupération de m bits $f_{I_t}^{i,j}$ qui constituent l'empreinte f_t. L'acheteur et le vendeur réalisent ensuite un protocole biparti pour insérer l'empreinte f_t dans le contenu I_t à l'aide d'une technique de tatouage de contenu W ($\mathsf{FI}_{\mathsf{B},\mathsf{I_t}} = \mathsf{W}(I_t^i, f_t)$). À la fin de cette phase, l'acheteur est le seul à obtenir une copie personnalisée $\mathsf{FI}_{\mathsf{B},\mathsf{I}}$ avec f_t du contenu alors que le vendeur apprend uniquement le *halfword* Hw. Ce *halfword* est stocké par le vendeur et utilisé pour tracer les acheteurs se conduisant de manière malveillante.

Lorsqu'une copie forgée $\mathsf{FI}_{\mathsf{I_t}}^*$ d'un contenu I_t est détectée (c'est-à-dire, le vendeur trouve une copie illégale d'un contenu), une phase d'*accusation* (Figures 1.10 et 1.11) est exécutée, dans laquelle un score $S_{1,t}$ est calculé pour chaque acheteur ayant acquis ce contenu. Ce score est calculé à l'aide du *halfword* Hw et de l'empreinte forgée f_*. Plus précisément, un score est calculé comme dans [128], qui reflète la probabilité que l'acheteur B_j a participé à la collusion qui a permis de forger la copie pirate. Si $S_{1,t}$ est supérieur au seuil d'accusation Z_{half1} pour le *halfword*, l'acheteur est déclaré suspect. Dans ce cas, son identité ainsi que le score d'accusation sont stockés dans une liste de suspect $\mathcal{LS}$. Sinon il est considéré comme innocent par le vendeur. Il faut noter que le vendeur ne connaît pas l'empreinte complète, mais seulement le *halfword* qui lui permet de calculer un score d'accusation partiel. Par la suite, le vendeur envoie $\mathcal{LS}$ au juge J. Le juge J (qui est une tierce partie de confiance) vérifie les calculs faits par le vendeur. Si l'un des calculs n'est pas correct alors le juge déclare le suspect innocent. Sinon, le juge ne faisant pas confiance au vendeur demande toutes les clefs permettant d'ouvrir

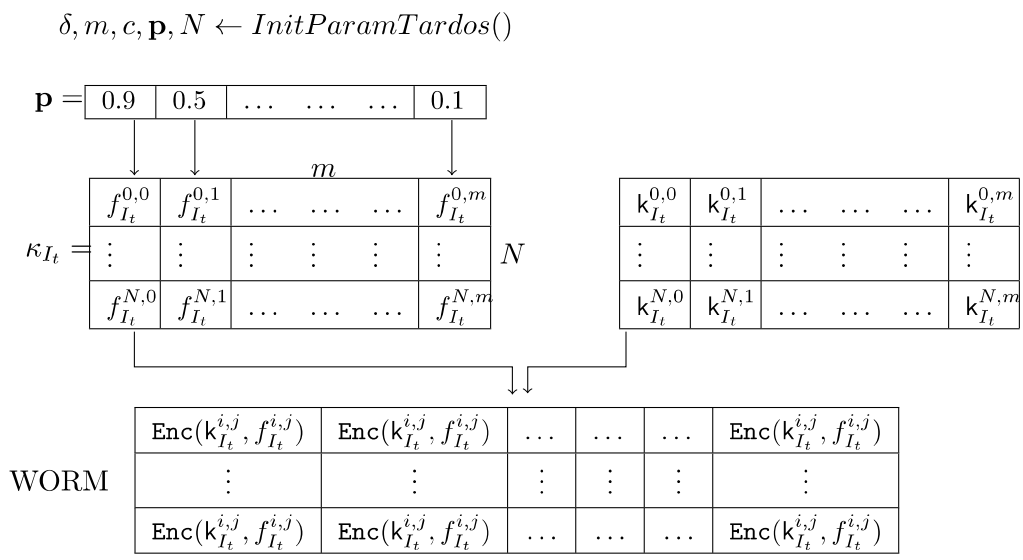

FIGURE 1.8.: Détail de la phase d'initialisation effectuée par le vendeur pour la solution de Charpentier *et co-auteurs*

le WORM pour obtenir les vraies probabilités $\mathbf{p}$. Puis, un second score est calculé sur l'empreinte complète pour prendre la décision finale. Ce second score $S_{2,t}$ est celui qui reflète vraiment la probabilité que l'acheteur soit coupable. Pour calculer ce score final, le *juge* force les acheteurs soupçonnés à révéler leurs empreintes complètes (un acheteur qui refuse de coopérer persuade le juge que cet acheteur est coupable). Ensuite, il réalise la même accusation, mais sur l'autre moitie de f_t (l'autre *halfword*), mais compare cette fois si le score $S_{2,t}$ avec le seuil d'accusation Z_{half2}. Si $S_{2,t}$ est supérieur à Z_{half2} alors le juge considère l'acheteur coupable de redistribution illégale.

Ici, le rôle du juge est double : 1) il vérifie que le vendeur a correctement calculé les scores partiels (c'est-à-dire, sans tricher), et 2) il calcule le second score requis pour procéder à l'accusation finale.

Kiayias, Leonardos, Lipmaa, Pavlyk et Tang. Kiayias, Leonardos, Lipmaa, Pavlyk et Tang ont proposé, dans [73] en 2015, un nouveau protocole de personnalisation de contenus asymétrique qui est aussi compatible avec les codes de Tardos et asymptotiquement optimal en terme de communication totale. Ce travail présente aussi deux nouvelles attaques qui peuvent affecter un protocole de personnalisation de contenus asymétrique basé sur les codes de Tardos. Celles-ci sont les suivantes : « retrait de l'accusation » et « arrêter et recommencer ». La première se présente dans la situation où le vendeur et le juge ne sont pas d'accord sur l'identité des acheteurs coupables. Avec les codes de Tardos, ceci peut arriver quand l'ensemble des acheteurs coupables identifiés par le vendeur et ceux par le juge n'ont aucun élément en commun. Nous discutons du fait que notre protocole n'est pas sensible à cette attaque dans la section 3.6.

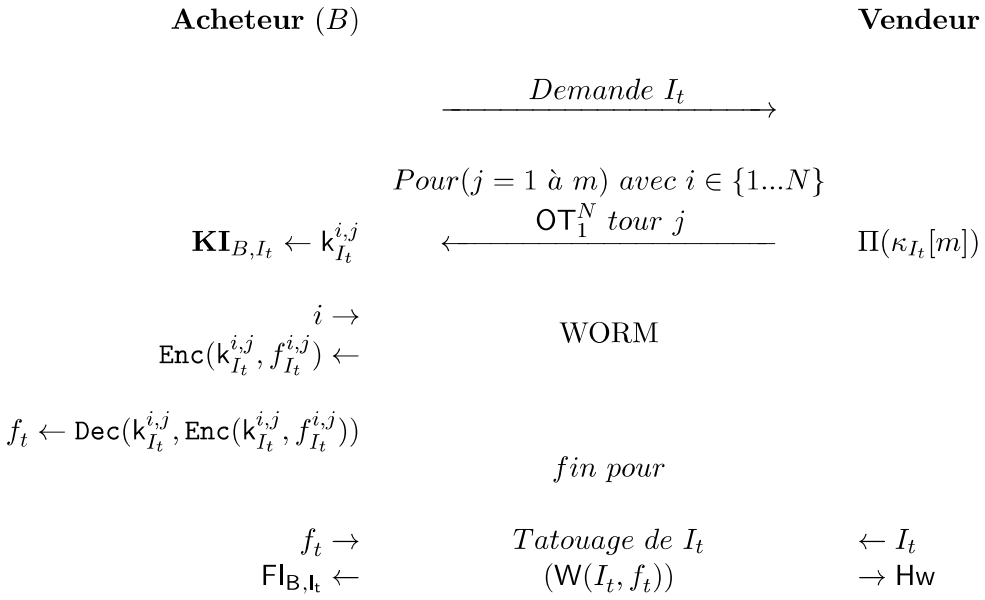

FIGURE 1.9.: Génération du mot de code et insertion de celui-ci dans le contenu de la solution de Charpentier *et co-auteurs*

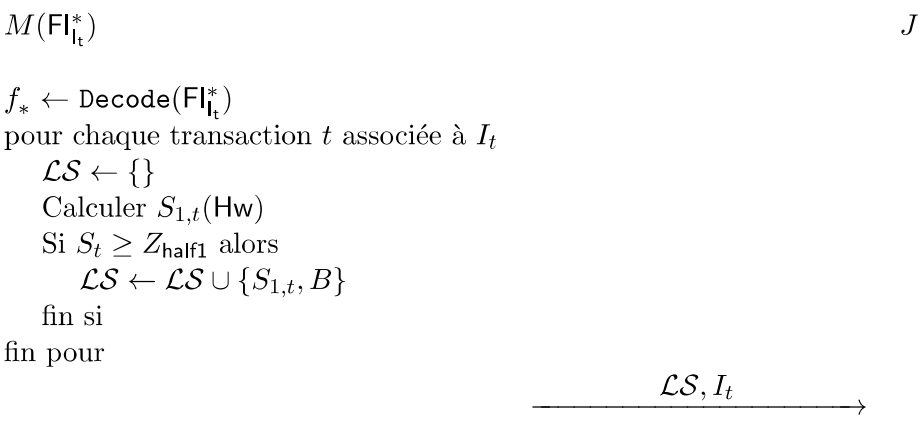

FIGURE 1.10.: Phase d'accusation de la solution de Charpentier *et co-auteurs* réalisée par le vendeur

$J(\mathcal{LS}, I_t)$ B_i

Si vérification réussie
pour chaque suspect dans $\mathcal{LS}$

$\xrightarrow{\quad Request\ \mathbf{KI}_{B,I_t} \quad}$

$\xleftarrow{\quad List\ of\ keys \quad}$

$f \leftarrow$ Déchiffrement des cellules du WORM avec $\mathbf{KI}_{B,I_t}$
$S'_k \leftarrow$ Calcul(f)
Si $Z_{\mathsf{half2}} \leq S'_k$
Accuse B_i
fin si
fin pour

FIGURE 1.11.: Phase d'accusation de la solution de Charpentier *et co-auteurs* réalisée par le juge

La dernière consiste à effectuer une séquence *arrêter et redémarrer* de la part d'un acheteur malveillant dont l'objectif est de l'aider à apprendre différentes versions des blocs avant l'arrêt complet du protocole. Les conséquences utiles d'une telle attaque sont les mêmes que celles d'une collusion avec, comme seule différence que l'attaque est menée par un seul acheteur malveillant. Nous expliquons dans la section 3.4 que notre protocole est résistant à cette attaque.

Dans le tableau 1.10 nous présentons les propriétés respectées par les solutions de personnalisation de contenus décrites précédemment. Comme nous le voyons, aucune de ces solutions ne respecte les propriétés de protection de la vie privée. Cela est normal, car ces solutions sont des protocoles asymétriques. De plus, aucune de ces solutions n'est compatible avec les codes anti-collusion de Tardos sans modifier de manière importante ces protocoles excepté les solutions [33] et [73] car ces dernières ont été conçues pour les utiliser. Aussi, aucune analyse formelle de sécurité n'est présentée. Pour finir, seule la solution proposée dans [73] a été implémentée, ce qui est rare dans la littérature.

1.7. Personnalisation de bases de données

Nous verrons la personnalisation de contenu dans le Chapitre 2, mais celle-ci ne peut pas être utilisée pour les bases de données. En effet, du fait de la structure spécifique des bases de données, il convient d'utiliser un moyen de personnalisation différent. C'est pour cela que des outils de personnalisation propres aux bases de données ont vu le jour. Le principal défi à relever est d'avoir les mêmes propriétés pour le tatouage de bases de données que pour le tatouage de contenu (robustesse, etc.). Cependant, les techniques matures dans la seconde catégorie ne peuvent être utilisées telles quelles dans la première. De plus, il est impératif de pouvoir tracer les traîtres ayant redistribué

	[105]	[107]	[92]	[33]	[73]
Anti-Framing	V	V	V	V	V
Traçage de Traître	oui	oui	oui	oui	oui
Analyse de sécurité formelle	X	X	X	X	V
Anonymat révocable	X	X	X	X	X
Non-chaînabilité des acheteurs	X	X	X	X	X
Non-chaînabilité des contenus	X	X	X	X	X
Compatible avec les codes de Tardos	X	X	X	V	V
Implémentation	X	X	X	X	V

Tableau 1.10.: Respect des propriétés de sécurité et protection de la vie privée par les protocoles présents dans la littérature. La couleur orange concernant le traçage de traîtres indique, soit que les codes de ces solutions sont trop grands pour être utilisés en pratique, soit que le code est inefficace contre les collusions. En vert, ce sont les codes utilisables en pratique et permettant de contrer des collusions.

illégalement une base de données acquise légalement. Certains fournisseurs de bases de données font signer un document indiquant que les usagers récupérant leurs bases de données n'ont pas l'autorisation de la redistribuer. Par exemple c'est le cas pour la base de données *MDCRS*[10] (*Meteorological Data Collection and Reporting System* – Accès le 16 février 2016) qui contient des informations météorologiques provenant de compagnies aériennes et dont les informations peuvent être obtenues par simple demande. Ces informations étant régies par le droit d'auteur, il est important de pouvoir tracer les traîtres si jamais certains utilisateurs ne respectent pas leurs engagements. La problématique majeure est qu'une fois redistribué, si celle-ci n'est pas personnalisée à l'aide d'une empreinte alors il est impossible de retrouver l'acquéreur responsable de la diffusion illégale. Nous présentons, dans cette section, les solutions principales de personnalisation de base de données présentes dans la littérature. Pour résumer, ces solutions utilisent principalement une clef secrète pour déterminer quels seront les bits des valeurs de certains attributs numériques de tuples spécifiques qui seront modifiés. Certaines solutions respectent des contraintes sur les données telles que la conservation de la moyenne pour un attribut donné.

Une image ou une vidéo est une suite de pixels disposés de manière spatiale et/ou temporelle et une chanson est une suite de fréquences. Il est possible de modifier le contenu original de manière invisible/inaudible. En effet, un œil humain ne voit pas chaque pixel d'une image, mais plutôt un ensemble de pixels tout comme l'oreille humaine n'est pas sensible à certaines fréquences ou encore ne distingue pas des fréquences proches. Ainsi, un nombre de modifications raisonnable permet de faire du traçage de traîtres sans trop dégrader l'image ou le son permettant de conserver l'utilité du support.

A contrario, dans une base de données, l'ajout de l'empreinte consiste en la modification de certaines valeurs d'attributs présents dans des tuples. Les valeurs modifiées

10. http ://amdar.noaa.gov/FAQ.html

ne doivent pas l'être au point de rendre la base de données inutilisable ni de fausser les conclusions des exploitations effectuées par l'acquéreur. De plus, dans une base de données, l'acheteur peut être amené à supprimer (volontairement ou non) des tuples possédant des valeurs d'attribut marquées. Il ne faut donc pas que l'acheteur soit capable de savoir quels tuples portent l'empreinte du tatouage.

Le premier article dans la littérature portant sur la personnalisation de bases de données fût proposé par Wagner, Fountain et Hazy en 1990 [132]. La personnalisation s'effectue sur des nombres et des dates en ajoutant ou retranchant une valeur δ fixe (positive ou négative) à l'attribut. Il y a donc deux valeurs possibles pour un attribut (la vraie valeur ou la valeur modifiée). Pour environ 50 pourcent des cas, la valeur de l'attribut sera soit la valeur originale soit la valeur originale moins delta. Dans les autres 50 pourcent, la valeur sera soit celle originale soit la valeur originale plus delta. Le choix de la valeur est déterminé par une fonction de hachage pseudo-aléatoire prenant en entrée la concaténation de l'identifiant de l'utilisateur, le nom de la relation, le nom de l'attribut et la valeur de la clef qui fournit un résultat binaire. Par exemple, si le résultat est 1 alors la valeur modifiée est utilisée sinon la valeur originale l'est. Les auteurs discutent des contraintes de certains attributs. Par exemple, en cas de modification de date, il faut tenir compte des week-ends, des vacances, pour que le résultat soit réaliste. Aussi, dans les données médicales, suivant l'utilité nécessaire des données à un instant précis alors les attributs pouvant être marqués ne sont pas les mêmes. En effet, certains étant critiques, leurs valeurs doivent rester inchangées (par exemple, les allergies d'un patient).

De manière générale, l'empreinte est insérée dans des attributs de type numérique en changeant le bit de poids faible. Cependant, comme mentionné précédemment, il est aisé de supprimer ou modifier des tuples. Il faut donc qu'il y ait suffisamment d'informations insérées. La restriction principale dans la personnalisation de bases de données est le fait que celle-ci doit rester utilisable et que les statistiques calculées ne doivent pas être faussées. De plus, contrairement à un contenu comme une image ou un film, une base de données est mise-à-jour régulièrement. Il ne faut donc pas que la mise-à-jour empêche de retrouver une marque dans une base de données diffusée.

Les solutions proposées dans la littérature respectent globalement le même modèle et fournissent donc les propriétés suivantes [13, 120, 42].

— *Détectabilité :* l'empreinte doit être détectable par le propriétaire en examinant les tuples.
— *Robustesse :* l'empreinte doit être robuste aux attaques (décrites plus en détail ci-après).
— *Mise à jour incrémentale :* la mise à jour d'une base de données ne doit pas supprimer l'empreinte.
— *Imperceptibilité :* les modifications effectuées lors de l'insertion de l'empreinte ne doivent pas rendre la base de données inutilisable.
— *Détection aveugle :* la base de données originale ne doit pas être nécessaire lors de la détection de l'empreinte. Bien que celle-ci ne soit pas nécessaire, il faut tout de même conserver des valeurs ayant permis de générer l'empreinte ou de l'insérer (clefs secrètes, etc), mais cela est moins coûteux que de garder la base complète.

— *Système basé sur les clefs :* Agrawal *et co-auteurs* suivent le principe de Kerckhoffs [72]. Ce principe dit que la sécurité du système doit reposer sur la clef qui paramètre le système de personnalisation et non sur l'obfuscation de la technique de personnalisation elle-même.

De plus, un certain nombre d'attaques ne sont réalisables que pour les bases de données [13, 120, 42].

1. *Mises à jour mineures :* l'adversaire peut mettre à jour des tuples de la base.
2. *Attaques malveillantes :*
   a) *Attaque sur les bits :* il est possible de changer certains bits de valeurs stockés dans la base de données. L'attaque la plus simple pour supprimer l'empreinte est de changer tous les bits. Cependant, cette manipulation entraînera forcément l'altération de la qualité de la base de données. L'attaque sur les bits peut être réalisée de trois manières différentes :
      i. *Attaque aléatoire :* à certaines positions, une valeur aléatoire est attribuée au bit.
      ii. *Attaque par zéro :* cette attaque consiste à mettre à zéro certains bits de certains attributs.
      iii. *Inversion de bit :* si le bit est à 0 l'adversaire met 1 et *vice versa.*
   b) *Attaque par arrondi :* cette attaque consiste à arrondir toutes les valeurs des attributs.
   c) *Attaque par sous-ensemble (ou sélection de sous-ensembles)* : la rediffusion ne se fait que sur un sous-ensemble de la base de données.
   d) *Attaque par mixage et correspondance (Attack Mix and Match en anglais) :* cette attaque consiste à remplacer des tuples par d'autres provenant d'autres bases de données et ayant des valeurs d'attributs proches.
   e) *Attaque additive :* l'adversaire ajoute sa propre marque dans la base de données. Cet ajout peut conduire à la destruction de certains bits de l'empreinte insérés par le propriétaire. Pour savoir à qui appartient vraiment la base de données, il suffit de regarder les empreintes. Si dans un tuple, les deux marques doivent se trouver au même endroit, mais avec une valeur de bit différente, la valeur du bit présente sera celle de l'empreinte insérée en dernier.
   f) *Attaque par inversion :* le but de cette attaque est de trouver une clef permettant de détecter l'empreinte. Cette clef peut être différente de la clef utilisée par le propriétaire.
   g) *Ajout de sous-ensemble :* l'adversaire ajoute des informations à la base de données en altérant peu sa qualité.
   h) *Altération de sous-ensemble :* cette attaque vise à modifier certains tuples.
   i) *Re-tri de sous-ensemble :* cette attaque a pour but de trier la base de données. Cette attaque ne peut fonctionner que si l'ordre des tuples est important lors de la personnalisation (modification/détection).

Dans la suite, considérons une base de données $\mathcal{D}$ possédant un ensemble de tuples $\mathcal{T}$. Un tuple est noté t et $t \in \mathcal{T}$. La clef primaire du tuple t est noté $\mathcal{P}$. Un attribut d'un tuple t est noté a et appartient à l'ensemble des attributs $\mathcal{A}$ ($a \in \mathcal{A}$).

Agrawal et Kiernan : Agrawal et Kiernan [13] ont proposé en 2002, une solution de personnalisation de bases de données. Dans cette solution, une clef secrète, connue uniquement du propriétaire, est utilisée pour déterminer quels attributs, à quel endroit et dans quels tuples ces attributs seront marqués [11]. Leur méthode est résistante contre certaines attaques (numéros 1 à 2.f) et leur article décrit les propriétés qui doivent être respectées (robustesse, exactitude, incrémentation, détection aveugle et caractère public de l'algorithme de personnalisation, principe de Kerchkoff).

Pour chaque tuple $t \in \mathcal{T}$, la clef primaire $\mathcal{P}$ du tuple t est concaténée avec la clef secrète sk avant d'être hachée à l'aide d'une fonction de hachage $\mathcal{H}$ ($\mathcal{H}_{p\mathsf{sk}} = \mathtt{Hash}(p||\mathsf{sk})$). Ensuite, une deuxième fonction de hachage est initialisée avec la concaténation de $\mathcal{H}_{p\mathsf{sk}}$ et la clef secrète ($\mathcal{H}_{\mathsf{sk}(p\mathsf{sk})} = \mathtt{InitHash}(\mathcal{H}_{p\mathsf{sk}}||\mathsf{sk})$). Cette fonction de hachage retourne un entier e. Si e modulo le nombre de bits m que possède l'empreinte f est égal à zéro, alors le tuple actuel est choisi pour accueillir l'empreinte, sinon il est ignoré. Ensuite, il faut déterminer quel attribut a du tuple accueillera un bit de l'empreinte f_i. On obtient l'indice en calculant e modulo le nombre d'attributs pouvant être marqués (c'est-à-dire, le nombre d'attributs numériques). Pour finir, le choix du bit qui sera modifié dans l'attribut est effectué en fonction de e *modulo* le nombre de bits de poids faible pouvant être marqués. La valeur de l'empreinte à cet emplacement est obtenue par le hachage $\mathtt{Hash}(\mathcal{P}||\mathsf{sk})$ de la concaténation de la clef primaire $\mathcal{P}$ avec la clef secrète sk. Si la valeur retournée par $\mathtt{Hash}(\mathcal{P}||\mathsf{sk})$ est paire alors le bit d'empreinte est 0 sinon 1.

La détection de l'empreinte s'effectue exactement comme l'insertion sauf qu'au moment d'insérer un bit il suffit de faire une lecture de celui présent et il est seulement nécessaire d'utiliser la base de données redistribuée pour cela. En effet, leur système possède la propriété de détection aveugle, ce qui signifie que la base de données originale n'est pas nécessaire pour retrouver l'empreinte. Compter le nombre total de correspondances entre le bit lu et le bit qui devrait se trouver dans l'empreinte (déterminé par l'algorithme de détection) permet de calculer un score d'accusation pour un individu. Plus le nombre de correspondances est élevé, plus l'individu est suspect.

Le fait qu'un tuple soit marqué ou non dépend de la clef primaire. De plus, il faut utiliser une clef secrète différente pour chaque divulgation de la base de données (c'est-à-dire, une pour chaque utilisateur à qui on la transmet) et la stocker pour pouvoir faire la détection. Dans cette solution à chaque fois que le propriétaire marque la base de données, il faut recalculer les emplacements possibles pouvant accueillir l'empreinte ce qui peut s'avérer coûteux notamment en terme de temps de calcul avant la distribution.

Sion, Atallah et Prabhakar : Sion, Atallah et Prabhakar ont publié un article en 2003 [120] permettant de marquer une base de données sous la supposition que les

11. Une supposition faite par Agrawal *et co-auteurs* est que l'algorithme est public

attributs sont de types numériques. L'utilisation qui sera faite des données est prise en compte en utilisant une métrique, nommée Erreur Quadratique Moyenne Maximum Permise (*Maximum Allowable Mean Squared Error* en anglais), qui leur permet de savoir si la base de données tatouée est conforme à l'utilisation qui en sera faite. Plus précisément, cette métrique sert à déterminer que la perte de qualité est acceptable. Leur méthode permet de contrer, dans cette proposition, les attaques 2.c et 2.g à 2.i mentionnées ci-dessus.

Pour marquer la base de données, des sous-ensembles uniques de tuples (et qui n'ont aucune intersection avec les autres sous-ensembles) sont sélectionnés. Pour cette sélection, la méthode effectue un tri reposant sur une fonction de hachage à sens unique qui prend en entrée une clef secrète. Sans cette clef, un adversaire ne pourra déterminer l'ordre des tuples utilisé lors de la personnalisation. La taille du sous-ensemble est un pourcentage du nombre de tuples de la base de données. Les sous-ensembles ne sont donc pas toujours de même taille. Ensuite, pour chaque sous-ensemble sélectionné, composé de tuples adjacents, et jusqu'à ce que la liste des sous-ensembles soit parcourue, un bit est inséré à plusieurs endroits du sous-ensemble avant de vérifier si l'utilité des données est bien préservée. Si, à l'aide de la métrique, les données ne sont pas utilisables, alors il faut essayer à nouveau de les marquer (toujours dans le même sous-ensemble) en utilisant d'autres paramètres pour l'encodage. Si les données ne sont toujours pas utilisables, alors ce sous-ensemble ne peut pas être marqué et est étiqueté comme non utilisable.

Dans le cas où le vendeur trouverait une copie pirate, il peut retrouver l'empreinte en commençant par extraire les sous-ensembles sélectionnés durant la première phase puis en extrayant tous les bits de l'empreinte dans tous les sous-ensembles. Pour retrouver l'acheteur, un schéma de vote par majorité qui permet de retrouver les voisins les plus proches de l'empreinte extraite est utilisé. Ce système est à détection aveugle.

Li, Swarup et Jajodia : Li, Swarup et Jajodia furent les premiers à proposer, dans [135], une solution utilisant des codes anti-collusion couplée à la solution décrite par Agrawal et Kiernan [13]. Plus précisément, les codes anti-collusion utilisés sont une adaptation de ceux proposés par Boneh et Shaw dans [22] (décrits dans la section 1.5).

L'empreinte insérée est composée de deux parties. La première est commune à tous les acheteurs ($f_1^{m_1}$ de taille m_1) et la seconde est unique et provient de la solution de [22] ($f_2^{m_2}$ de taille m_2). La taille de l'empreinte complète est $m = m_1 + m_2$. L'insertion de l'empreinte se fait quasiment comme dans la solution de Agrawal *et co-auteurs*. En effet, pour cacher la distribution des bits de l'empreinte, un masque est utilisé sur chaque bit de cette dernière et le bit inséré est en fait le *XOR* entre le bit de f et le masque. Du fait de la *marking assumption*, lors d'une collusion les acheteurs généreront une copie de la base de données contenant $f_1^{m_1}$. De plus, la permutation, présente dans [22], n'est plus nécessaire et au lieu de générer aléatoirement l'empreinte, celle-ci est générée pseudo-aléatoirement en utilisant la clef privée et le numéro de l'acheteur comme graine de la fonction pseudo-aléatoire, ce qui rend l'empreinte déterministe pour le vendeur.

La détection de l'empreinte se fait comme dans Agrawal *et co-auteurs* (modulo l'utilisation du masque). Cependant, lors de la détection, la première étape est de vérifier

$f_1^{m_1}$. Si un bit n'est pas correct, l'algorithme ne peut pas retrouver de suspect.

Cependant, contrairement à la solution proposée par Gross-Amblard *et co-auteurs* présenté ci-après, les auteurs de [135] ne respectent pas de contraintes sur la base de données. Aussi, la supposition que $f_1^{m_1}$ est non modifié lors de l'attaque est une hypothèse forte. En effet, si avant de faire une collusion les acheteurs réalisent une attaque sur la base de données alors certains bits de $f_1^{m_1}$ peuvent potentiellement être modifiés ou disparaître. Avec la méthode proposée si un bit de $f_1^{m_1}$ est manquant alors aucun traître ne peut-être identifié ce qui est un inconvénient majeur.

Gross-Amblard *et co-auteurs* : David Gross-Amblard [63] *et co-auteurs* dans [42, 79] proposent une solution un peu différente. Dans [79], une phase de pré-calcul est d'abord effectuée dans le but de déterminer les bits de poids faibles b d'un attribut A d'un tuple $t \in \mathcal{T}$, qui recevront l'empreinte. Ces positions sont stockées par le vendeur. Marquer les bits b permet de ne pas trop perturber les données pour pouvoir conserver l'utilité de celles-ci. Ensuite, il suffit de marquer la base en utilisant une clef secrète différente pour chaque utilisateur au moment de la distribution.

Cette méthode, basée sur [13] et [120], est améliorée pour préserver le résultat des requêtes (*query-preserving watermarking* en anglais) sur la base de données. Pour cela, un langage déclaratif est spécifié. Ce langage permet de déclarer les contraintes que devront respecter les bases de données une fois marquées. Ces contraintes sont de deux types : linéaires et générales. Pour le premier type, les contraintes sont traduites dans un *ILP* (*integer linear program* en anglais ou programme linéaire d'entier). Pour le second la solution qui est un algorithme glouton nommé *Greedy method* en anglais ou *méthode gourmande*, de Sion *et co-auteurs* [120] est utilisée. Cependant, le nombre de requêtes possibles sur la base de données est limité à celles prévues par leur modèle et le placement de l'empreinte est plus difficile.

L'avantage de cette technique, comparée aux méthodes [13] et [120], est qu'elle est compatible avec les codes de Tardos. Cela apporte une meilleure résistance aux collusions. Comparé à [135], les contraintes linéaires et générales sont respectées. De plus, même s'il manque des bits de l'empreinte cela n'empêche pas d'accuser un suspect. Enfin, cette solution étant aveugle, il est donc nécessaire de stocker uniquement les informations relatives à la génération de l'empreinte et à son insertion et non la base de données marquée.

La résistance, des solutions de tatouage présentées précédemment, contre les attaques décrites ci-dessus est présentée dans le tableau 1.11.

Attaques	[13]	[120]	[135]	[63]
Mises à jour mineures	$\mathcal{V}$	$\mathcal{X}$	$\mathcal{V}$	$\mathcal{V}$
Attaque sur les bits	$\mathcal{V}$	$\mathcal{X}$	$\mathcal{V}$	$\mathcal{V}$
Attaque par arrondi	$\mathcal{V}$	$\mathcal{X}$	$\mathcal{V}$	$\mathcal{V}$
Attaque par sous-ensemble	$\mathcal{V}$	$\mathcal{V}$	$\mathcal{V}$	$\mathcal{V}$
Attaque par mixage et correspondance	$\mathcal{V}$	$\mathcal{X}$	$\mathcal{V}$	$\mathcal{V}$
Attaque additive	$\mathcal{V}$	$\mathcal{X}$	$\mathcal{V}$	$\mathcal{V}$
Attaque par inversion	$\mathcal{V}$	$\mathcal{V}$	$\mathcal{V}$	$\mathcal{V}$
Ajout de sous-ensemble	$\mathcal{X}$	$\mathcal{V}$	$\mathcal{X}$	$\mathcal{X}$
Altération de sous-ensemble	$\mathcal{X}$	$\mathcal{V}$	$\mathcal{X}$	$\mathcal{X}$
Re-tri de sous-ensemble	$\mathcal{X}$	$\mathcal{V}$	$\mathcal{X}$	$\mathcal{X}$
Collusion	$\mathcal{X}$	$\mathcal{X}$	$\mathcal{V}$	$\mathcal{V}$

Tableau 1.11.: Résumé des travaux précédents se protégeant des attaques décrites dans la section 1.7

2. Respect de la vie privée

Depuis la démocratisation d'Internet, de plus en plus de services sont proposés par des entreprises, qui parfois se permettent de collecter un nombre important de données personnelles. Ainsi, l'utilisation du GPS sur un téléphone intelligent indique, par exemple, les déplacements et centres d'intérêt d'un utilisateur, l'utilisation d'une carte de fidélité révèle les achats effectués, celle d'une carte bancaire indique où ces achats ont été effectués. Il s'agit de quelques-uns des moyens existants pour récupérer des informations sur les utilisateurs. Ainsi, les utilisateurs de technologies de l'information et de la communication sont constamment tracés et la récolte d'informations a tendance à se généraliser souvent à leur insu. De plus, le suivi s'effectue, à plus grande échelle, par des entités gouvernementales (par exemple, la NSA avec le programme PRISM, la France a la loi Renseignement). Toutes ces informations, qu'elles soient données volontairement ou non, peuvent causer une atteinte à la vie privée des individus. Par exemple, même lorsque l'identité d'un usager n'est pas connue de prime abord, il est parfois possible de la déduire à partir des informations collectées. Il est donc primordial de limiter la collection de ce type d'information afin d'en limiter les potentiels usages malveillants.

Dans ce chapitre, nous aborderons tout d'abord les principes généraux de la vie privée et de sa protection, les lois existantes et le respect de la vie privée dans la société de l'information (2.1). Nous parlerons ensuite des menaces et des attaques contre la vie privée (section 2.2). Dans une troisième section (section 2.3), nous présenterons les mécanismes permettant d'assurer la protection de la vie privée. Dans la section 2.5 nous présenterons le paradoxe entre l'identification et le respect de la vie privée et nous justifierons l'importance de ce sujet de recherche. Nous conclurons ce chapitre, en présentant l'état de l'art concernant la personnalisation de contenus préservant la vie privée des acheteurs dans la section 2.4.

2.1. Respect de la vie privée : principes, cadre légal et société

La vie privée est protégée par des lois (françaises, européennes, etc.), mais elle n'est pas définie clairement. En effet, comme nous le verrons par la suite dans le cadre légal, les lois sont définies de manière assez vague. De plus, l'apport régulier de nouvelles technologies constitue un risque pour le respect de la vie privée

2.1.1. Principes

Le respect de la vie privée est un droit fondamental pour tout individu, comme l'indique l'article 9 du Code civil [1] où, « chacun a droit au respect de sa vie privée »,

ainsi que dans la Déclaration des droits de l'Homme de 1948 à travers l'article 12 [8] : « Nul ne sera l'objet d'immixtions arbitraires dans sa vie privée, sa famille, son domicile ou sa correspondance [...]. Toute personne a droit à la protection de la loi contre de telles immixtions ou de telles atteintes. »

Le droit au respect de la vie privée est, d'après la jurisprudence « le droit pour une personne d'être libre de mener sa propre existence avec le minimum d'ingérences extérieures » ([1] partie 1 : « La définition du "droit au respect de la vie privée" »). En d'autres termes, une personne a le droit de conserver sa vie privée pour elle-même et de faire ce qu'elle veut dans sa vie privée sans subir d'intrusion extérieure sur celle-ci. Ce droit protège, entre autres, son nom, son image, son intimité.

Dans le dictionnaire de l'Académie française, la vie privée est définie comme étant ce « qui est relatif à la vie personnelle de chacun ; individuel, personnel, intime. » [2]. Elle est donc constituée de toutes informations personnelles d'une personne (identité, goûts, habitudes, âge, opinions politiques, orientation sexuelle, croyance religieuse) qui doivent rester en dehors de la vie publique, sauf dans les cas définis par la loi.

De plus, tout individu que ce soit sur Internet ou dans le monde réel laisse des traces numériques, et ce tous les jours. Une trace numérique est le résultat d'une action telle qu'un paiement par carte bancaire, une navigation sur des sites internet ou encore l'utilisation de cartes de bus. Toutes ces traces relatent ce qu'un individu aime, mais aussi ses habitudes, etc. Ce sont donc, par définition, des données à caractère personnel qu'il est important de protéger.

Protéger la vie privée d'un individu peut donc s'apparenter à minimiser les traces générées et au contrôle sur les données personnelles qu'il dissémine. Ainsi, une donnée à caractère personnel est constituée, d'après la loi « informatique et libertés » article 2 [17] de toute information permettant d'identifier de manière directe (nom) ou indirecte (numéro de téléphone, numéro de sécurité sociale) un individu. Il est donc nécessaire de vérifier si une personne est identifiable par l'utilisation de moyens pouvant être possédés par toute personne. En effet, certaines informations ne portent pas atteinte à la vie privée (genre, âge, code postal) si elles sont prises séparément, mais qui, lorsque celles-ci sont mises ensemble permettent d'identifier un individu de manière unique. Pour la CNIL [12] « les données sont considérées, à caractère personnel, dès lors qu'elles concernent des personnes physiques identifiées directement ou indirectement ». Ces définitions, article 2 de la loi informatique et libertés et celle de la CNIL, sont très larges et couvrent un grand nombre de données. En effet, les données permettant l'identification ne sont pas énumérées. En France, il est aussi possible de demander la suppression de toute information concernant une personne sur demande de sa part. Cette demande doit être motivée et justifiée. Cependant, pour certaines organisations comme les services publics (par exemple le centre des impôts), il est évident que cela est impossible, car cette organisation a un motif légitime pour traiter ces données. Pour protéger la vie privée d'un individu, il faut tenir compte de son identité directe (nom, prénom), des identifiants le concernant (numéro de sécurité sociale, etc.), mais aussi de ce que l'on appelle *quasi-identifiants* qui se composent d'une combinaison d'attri-

12. cil.cnrs.fr/CIL/spip.php ?rubrique299 - accessible le 01-09-2015

buts qui, pris séparément, semblent anodins, mais qui forment une combinaison unique lorsqu'ils sont mis ensemble. Toutes ces données sont collectées par une entité nommée *collecteur*.

Le *respect* de la vie privée est régi par les six principes suivants [70].

1. *Minimisation des données :* le collecteur de données ne doit demander que celles nécessaires à son service.
2. *Souveraineté des données :* les données appartiennent exclusivement à l'individu qu'elles concernent, il doit donc pouvoir contrôler ce que le collecteur fait de ces informations.
3. *Consentement explicite :* l'utilisateur doit exprimer de manière explicite son consentement pour la collecte des données personnelles le concernant (par exemple, en cochant la mention « *Je comprends et j'accepte les conditions de collecte des données* »).
4. *Transparence :* l'utilisateur doit pouvoir se faire expliquer comment le système gère ses données afin d'éviter de lui faire confiance de manière aveugle. Ainsi, si le système est une boîte noire, l'utilisateur ne saura pas comment ses données sont gérées et n'aura donc pas de raisons objectives de lui accorder sa confiance.
5. *Imputabilité :* le système collectant les données doit mettre en œuvre les moyens nécessaires pour sécuriser les données des utilisateurs (par exemple, à l'aide des mécanismes de contrôle d'accès et de cryptographie). S'il ne le fait pas, il peut être tenu pour responsable dans le cas d'une compromission de la vie privée des utilisateurs.
6. *Droit à l'effacement :* tout individu peut demander la suppression de ses informations personnelles collectées par le système. Cela peut, par exemple, se produire après avoir eu une carte de fidélité dans un magasin. Il faut dans ce cas demander à ce que toutes les informations soient effacées du système du magasin, en fournissant un motif valide. L'entité collectrice est obligée par la loi de le faire, même s'il existe des exceptions. En effet, un individu ne peut pas demander au service des impôts de supprimer toutes les informations le concernant.

Aussi, les autorités de protection des données européennes considèrent toutes que certaines données telles que l'adresse IP d'une personne est une donnée à caractère personnel alors que la Cour d'appel de Paris considère que « les adresses IP collectées à l'occasion de la recherche et de la constatation des actes de contrefaçon sur Internet ne permettent pas d'identifier, même indirectement, des personnes physiques et que, dès lors, elles ne constituent pas des données à caractère personnel. » [9]

Modèle d'adversaire Les données personnelles des utilisateurs sont des informations à forte valeur économique. En effet, les traces numériques collectées peuvent servir à faire du profilage (par exemple, pour un site de commerce électronique). On peut définir un *adversaire* comme toute entité collectant des données personnelles sur un individu dans le but de commettre une action frauduleuse (usurpation d'identité, etc.). Dans la littérature, il existe trois types d'adversaires contre la vie privée. Ces adversaires

sont proches de ceux utilisés en cryptographie et en sécurité. Ils se différencient en termes de capacité de nuisance, des ressources auxquelles ils ont accès (capacité de calcul ou de mémoire) ou encore des connaissances qu'ils possèdent. Le premier type d'adversaire *honnête-mais-curieux* suit le protocole (transmission de message, paiement d'une transaction, etc.), mais enregistre tout ce qu'il voit. Le protocole est considéré comme sécurisé si, dans ce contexte, la vie privée n'est pas brisée malgré la connaissance des informations enregistrées. Le deuxième adversaire est un adversaire *malveillant*, qui triche activement durant le protocole pour le faire dévier et apprendre des informations (déchiffrement des messages) ou tout simplement le faire échouer. Le dernier adversaire est un *espion* qui a la capacité de surveiller des liens de communications entre deux nœuds d'un réseau, mais ne peut pas corrompre ou influencer ces nœuds. La différence entre un adversaire honnête-mais-curieux et un espion réside dans le fait que l'adversaire honnête-mais-curieux est interne au protocole (c'est-à-dire, une entité de celui-ci) tandis que l'espion est externe au protocole.

2.1.2. Cadre légal

De nombreuses lois existent pour protéger la vie privée des utilisateurs, que ce soit au niveau national, européen ou international. Au niveau français, la CNIL (Commission Nationale de l'Informatique et des Libertés) est en charge d'assurer que l'informatique sert les individus et ne nuit pas à ceux-ci (identité, droit de l'homme, vie privée, liberté individuelle) [13]. La CNIL est une autorité administrative indépendante. Comme on peut le voir, il n'existe pas de loi précisant quelles sont les informations considérées comme étant à caractère personnel. De plus, avant toute collecte de données personnelles, le collecteur doit faire une demande auprès de la CNIL pour obtenir une autorisation.

Au niveau européen, la directive 95/46/EC [5] est le texte de loi fondamental pour la protection des données à caractère personnel. Ce texte définit les principes de proportionnalité (*minimisation des données* de la section précédente), de transparence et de finalité légitime (nommée *transparence* et *consentement explicite* dans la section précédente). En effet, toute collecte d'informations à caractère personnel doit être approuvée par l'utilisateur, qui doit être informé de la finalité de la collecte (son but) et cela en toute transparence (c'est-à-dire, en étant informé de l'utilisation qui sera faite de ses données et comment elles seront disséminées). De plus, pour la finalité déclarée seules les informations nécessaires doivent être collectées (principe de minimisation). En Europe, les données à caractère personnel ne doivent pas (sous peine de poursuites) quitter l'Union européenne sauf dans certains cas spécifiques (principe de souveraineté des données). Par exemple, « si le transfert a lieu vers un pays reconnu par la Commission européenne comme « Offrant un niveau de protection des données suffisant » ».

De plus, la Convention Européenne des Droits de l'Homme indique dans l'article 8 [3] intitulé « Droit au respect de la vie privée et familiale » que « *Toute personne a droit au respect de sa vie privée et familiale, de son domicile et de sa correspondance. Il ne peut y avoir ingérence d'une autorité publique dans l'exercice de ce droit pour autant*

13. cnil.fr/linstitution/qui-sommes-nous/ – accessible le 27-07-2015

que cette ingérence est prévue par la loi [...] *nécessaire à la sécurité nationale, à la sûreté publique,* [...] *ou à la protection des droits et libertés d'autrui.* ≫. De plus, la Commission Européenne [7] oblige tout collecteur de données personnelles à informer tout utilisateur du système informatique qu'il collecte des données personnelles, en indiquant son nom, la manière dont cette collecte est effectuée ainsi que l'endroit où les données seront transférées. Le collecteur doit aussi informer que l'utilisateur peut à tout moment demander la suppression de ces données, ou recevoir une copie des données collectées le concernant. Enfin, la commission européenne explique que les collecteurs doivent respecter certaines obligations [6]. À titre d'exemple, nous pouvons citer que les informations doivent être collectées dans un but légitime et explicite et aussi que les données collectées doivent être protégées de manière appropriée (en les sécurisant suffisamment : principes d'imputabilité).

Une autre directive, 97/66/CE, existe dont l'objectif est d'obliger le secteur des télécommunications à tenir compte des évolutions technologiques pour offrir un meilleur niveau de sécurité. Cette dernière ajoute à la directive 95/46/CE des règles spécifiques au secteur des télécommunications pour offrir un niveau de protection aux usagers équivalent à celui présent dans la directive 95/46/EC, quelle que soit l'évolution des technologies et des marchés. De plus, la directive 2002/58/CE ajoute que les méthodes de traçage des utilisateurs (pixels invisibles, identificateurs et autres dispositifs etc) doivent être portées à l'attention de l'utilisateur si elles sont utilisées, et doivent uniquement être mises en place dans des cas légitimes. Pour finir, la Directive 2006/24/CE [4] est une mise à jour de la précédente et ajoute l'obligation suivante : les fournisseurs de services doivent conserver pour une durée minimale de six mois et maximale de deux ans toutes informations permettant d'identifier la source et la destination d'une communication, et les informations permettant de localiser les sources et destinations. Ceci est fait ≪ en vue de la prévention, de la recherche, de la détection et de la poursuite d'infractions pénales ≫. Bien que cela soit compréhensible, et autorisé par l'article 9 qui indique qu'≪ en vertu de l'article 8 de la Convention Européenne de sauvegarde des droits de l'homme et des libertés fondamentales (CEDH), toute personne a droit au respect de sa vie privée [...] Il ne peut y avoir ingérence d'une autorité publique dans l'exercice de ce droit que pour autant que cette ingérence est prévue par la loi [...] à la sécurité nationale, à la sûreté publique, à la défense de l'ordre et à la prévention des infractions pénales, ou à la protection des droits et des libertés d'autrui. [...] L'adoption d'un instrument relatif à la conservation des données constitue dès lors une mesure nécessaire au regard des exigences de l'article 8 de la CEDH. ≫. Cet article n'en est pas moins une menace contre la vie privée des individus, car il autorise la collecte et le stockage d'informations bien qu'il limite le stockage dans le temps. De plus, il est très difficile de vérifier si ces données ont bien été supprimées et de s'assurer qu'elles ne sont pas conservées.

2.1.3. Respect de la vie privée dans la société de l'information

Nous avons vu dans la section précédente que la protection de la vie privée est encadrée par un cadre légal. Cependant, ce cadre légal est insuffisant, car la vie privée

est bien souvent brisée dans le monde réel. En effet, le scandale PRISM, les moteurs de recherche, les réseaux sociaux ainsi que d'autres avancées technologiques collectent de grandes masses de données qui sont souvent non nécessaires à leur fonctionnement. Dans cette section, nous survolerons brièvement le problème soulevé par certaines de ces technologies.

PRISM La NSA (Agence Nationale de Sécurité ou *National Security Agency* en anglais) est une agence gouvernementale américaine ayant pour but de surveiller toutes les communications étrangères sur le sol américain et de protéger les communications de ses concitoyens. Cependant, en juin 2013, des documents confidentiels ont été divulgués par Edward Snowden. Ces documents ont révélé que la NSA a collecté des informations sur ses concitoyens, ce qui est interdit dans le droit américain, ainsi que sur les alliés des États-Unis d'Amérique (notamment européens) dont les utilisateurs d'Apple, de YouTube, de Google, de Facebook, de Microsoft et de Yahoo. Pour ce faire, depuis 2007, ils ont utilisé le logiciel PRISM qui permet de surveiller toutes les communications numériques échangées. D'après certaines révélations [124], PRISM était connecté directement aux serveurs d'entreprises tels que Google ou Facebook qui ont démenti ces informations. Dans tous les cas, les entreprises américaines doivent coopérer avec la NSA en divulguant des informations sur des utilisateurs étrangers sur la demande de celle-ci ou de toute autre agence gouvernementale, du fait de l'existence du *Patriot Act*. La presse a beaucoup parlé de PRISM, mais il est plausible que toutes les puissances mondiales aient des programmes ressemblant à celui-ci (Allemagne [82], France [27], etc.).

Loi Renseignement Ayant pour but d'accroître la sécurité nationale française, la loi Renseignement autorise et encadre l'action des services de renseignement français lorsqu'ils souhaitent espionner les communications, notamment dans le cadre de la lutte contre le terrorisme. De par ces mesures, ce texte de loi est très critiqué notamment à cause des dangers et des difficultés techniques liés à l'application de cette loi [27, 11] qui mettent en avant les dangers et les difficultés techniques liés à l'application de cette loi. En particulier, les algorithmes de détection utilisés fouillent massivement les données (plus spécifiquement les métadonnées) pour extraire de potentiels comportements terroristes avant d'identifier ces personnes et de réaliser une recherche plus approfondie. Auparavant, la situation était complètement différente puisque la loi n'autorisait les fouilles approfondies qu'au cas par cas sur un ensemble restreint d'utilisateurs. La CNIL a émis un avis sur une version de loi renseignement. Cette dernière fût par la suite légèrement modifiée. Parmi les modifications, nous pouvons citer que toutes les données interceptées en temps réel sur les réseaux des opérateurs ne doivent être constituées exclusivement que de métadonnées de connexion (et donc ne pas concerner le contenu). De plus, la durée de conservation des données a été raccourcie. Cependant, la CNIL n'aura aucun pouvoir dans ce contexte et ne pourra donc pas effectuer les vérifications nécessaires, notamment sur le fait que les données recueillies sont bien détruites après la durée de conservation déclarée. De plus, certaines pratiques autorisées dans la loi Renseignement étaient déjà réalisées illégalement ce qui laisse à penser que la loi n'est

faite que pour légaliser des pratiques existantes. Pour finir, la collecte de métadonnées permet de connaître plus d'informations sur la vie privée des utilisateurs que le contenu lui-même ce qui est extrêmement invasif pour la vie privée.

Google Google est une entreprise qui conçoit de nombreux produits. En effet, Google est le premier moteur de recherche et le système d'exploitation Android est utilisé sur 81,5% des smartphones [14] et 67,7% des tablettes [15] vendus en 2014. De plus, avec des produits comme Gmail, Google+, Google Maps ou encore Navigation, Google collecte une quantité importante de données personnelles. Par exemple, si on considère Gmail, le fait d'avoir accès à des informations telles que tous les e-mails, la liste des contacts, permet à Google de réaliser des statistiques pour savoir avec qui nous communiquons le plus ou encore d'analyser les e-mails pour de la publicité ciblée. De plus, certaines applications Android développées par Google (ainsi que plus généralement toutes les applications disponibles pour Android) demandent l'accès à des informations, stockées sur le téléphone, dont elles n'ont pas nécessairement besoin. Enfin, le navigateur Chrome envoie les références de toutes les pages internet consultées par les utilisateurs à Google, qu'ils soient ou non utilisateurs du moteur de recherche Google. Ceci a pu se faire à l'aide du service *Google Suggest* et avait pour finalité déclarée de mieux connaître le comportement des utilisateurs [102]. Comme Google transmet les informations qu'il détient au gouvernement américain à cause du *Patriot Act*, on peut considérer que ce dernier a aussi accès à ces informations.

Les réseaux sociaux Plusieurs milliards de personnes utilisent les réseaux sociaux (tels que Facebook ou Twitter) à travers le monde et partagent en grande quantité leurs données personnelles (lieu d'habitation, date de naissance, activité, photo). Ces informations personnelles peuvent servir à des personnes malveillantes dans le but de voler l'identité d'une personne, de cambrioler une habitation [16] ainsi que pour d'autres faits malveillants. En effet, prenons l'exemple d'une personne partant en vacances et l'indiquant à tout le monde de manière publique, à l'aide d'un message sur Twitter ou Facebook. À l'aide de son nom, il peut être possible de trouver son adresse dans les pages jaunes et, sachant que cette personne n'est pas présente chez elle, aller la cambrioler. Il est donc important, en plus de vouloir protéger la vie privée des utilisateurs, de les sensibiliser aux dangers du partage d'informations par des messages ou des photos, ainsi qu'aux risques liés au fait de laisser son profil social ouvert (c'est-à-dire, lisible par n'importe qui). Aussi à titre d'exemple, en 2013, il était possible de suivre les déplacements de 35% [17] des 18-34 ans aux États-Unis sur Foursquare. De plus, un autre danger des réseaux sociaux est la possibilité qu'a un utilisateur de partager les

14. Chiffre divulgué par ZDNet : zdnet.fr/actualites/chiffres-cles-les-os-pour-smartphones-39790245.htm - Accessible le 01-09-2015

15. Chiffre divulgué par ZDNet : zdnet.fr/actualites/chiffres-cles-le-marche-des-tablettes-par-os-39790133.htm - Accessible le 01-09-2015

16. voir le site internet : pleaserobme.com/why - Accessible le 01-09-2015

17. marieclaire.fr/,facebook-twitter-les-reseaux-sociaux-allies-des-cambrioleurs,705961.asp - Accessible le 01-09-2015

données personnelles d'un ami, d'une connaissance, sans que celui-ci en soit conscient et donc sans qu'il puisse contrôler ce qui est divulgué. Ceci est une atteinte directe au principe de souveraineté.

Avancées technologiques et respect de la vie privée Les avancées technologiques telles que les téléphones intelligents, les télévisions connectées, les achats en ligne, etc. ont apporté leurs lots de dangers pour la vie privée des utilisateurs. En 2009, l'expérimentation de nouveaux compteurs d'électricité, d'eau et de gaz dits compteurs intelligents [10] a commencé. Les compteurs intelligents servent à mesurer de manière fine la consommation d'une maison ou d'une entreprise tout en la transmettant potentiellement en temps réel ou de manière régulière au gestionnaire du réseau. Le but déclaré est de pouvoir suivre de manière plus fine la demande, par exemple en électricité, afin de pouvoir mieux distribuer l'électricité sur le réseau. Le problème majeur de ces compteurs est l'inférence potentielle d'informations personnelles, faisant partie de la vie privée, réalisable à partir des informations collectées. En effet, la consommation énergétique d'un foyer permet de déduire de nombreuses informations sur les personnes (horaire de travail, heure de réveil et de coucher, présence ou absence dans l'habitation, etc.). Cette collecte d'informations a fait réagir la CNIL du fait que les informations envoyées sont des informations personnelles. Elle a ainsi déclaré : « Les informations de consommation d'énergie transmises par les compteurs sont très détaillées et permettent de savoir beaucoup de choses sur les occupants [...] Les distributeurs d'énergie devront donc apporter des garanties sérieuses sur la sécurisation de ces données et leur confidentialité. ». Les fournisseurs de services ont répondu que les données envoyées seraient chiffrées, mais cela ne résout pas le problème de vie privée, car les informations sont quand même transmises toutes les 10 à 30 min et celles-ci sont stockées en clair pour pouvoir être analysées.

2.2. Menaces et attaques contre la vie privée

Cette section est consacrée, tout d'abord dans une sous-section 2.2.1, aux menaces contre la vie privée telles que le traçage et le profilage d'utilisateurs. Puis, nous présenterons dans une deuxième sous-section 2.2.2 les attaques par inférence, par chaînage et par composition brisant la vie privée.

2.2.1. Les menaces contre la vie privée

Profilage Le profilage d'utilisateurs est extrêmement répandu sur Internet. Celui-ci consiste à créer un profil d'utilisateur contenant des informations personnelles. Ce n'est pas une attaque à proprement parler, car le but du profilage n'est pas de briser les mécanismes de protection de la vie privée, mais c'est une menace qui porte atteinte à celle-ci. Les principaux usages du profilage sont le ciblage publicitaire, la recommandation de contenu, etc. Ce processus est automatisé dans le sens où une intervention humaine n'est pas requise. Les informations collectées sont, en fait, des traces numériques récupérées par les moteurs de recherche, les sites de commerce électronique, les sites

internet en général, lorsqu'une carte à puce est utilisée (par exemple, carte de bus), par les réseaux sociaux, etc. Chaque information prise séparément n'apporte rien, mais le fait de les regrouper permet d'obtenir un profil de chaque utilisateur contenant ses habitudes, ses goûts, etc. De plus, le profilage peut avoir comme conséquence une discrimination sur le prix qui peut être déterminé à l'aide du profil. Le profil peut contenir, par exemple, le nombre de fois où vous consultez la page d'un contenu sur un site de commerce électronique et faire ensuite, à l'aide d'outils automatisés, une analyse comportementale. Par exemple, en 2000 Amazon a été accusé de faire de la discrimination sur le prix [102]. Celle-ci consiste à moduler les prix des contenus en fonction des profils des utilisateurs créés par Amazon. En effet, il arrive qu'Amazon propose un contenu à un certain prix lors d'une consultation par un utilisateur sur un ordinateur A et un autre prix au même utilisateur surfant à partir d'un ordinateur B différent de A. Google a annoncé en 2009, qu'il utilisait le ciblage publicitaire à l'aide de profils dynamiquement construits possédant 600 centres d'intérêt, ce qui permet d'obtenir un profil très précis de chaque utilisateur [18]. Il est possible de générer un profil d'utilisateur grâce aux techniques de fouille de données (ou *Data Mining* en anglais). En effet, le but de la fouille de données est d'extraire des informations pertinentes d'un ensemble important de données récoltées telles que des comportements et des habitudes. Nous ne rentrerons pas dans les détails sur la fouille de données qui est en dehors de la thématique de cette thèse. Cependant, un lecteur désirant en savoir plus peut se reporter à l'article proposé par Hand et Mannila et Smyth [68].

Traçage La seconde menace concerne le traçage d'utilisateurs, ce qui revient à suivre leurs activités. Il existe plusieurs objectifs pour lesquels le traçage d'utilisateurs est utilisé par exemple, pour apprendre ce qui intéresse les utilisateurs d'un site web, car ceci est pertinent pour le propriétaire du site (par exemple, pour de la recommandation de produit, du ciblage publicitaire, etc.). Également, connaître les habitudes ou la localisation d'un utilisateur peut être utile pour recommander un lieu. Le traçage s'effectue en général à l'aide de cookies, mais peut être effectué par les boutons « j'aime » et « +1 » même si l'utilisateur n'est pas connecté à son compte de réseaux sociaux et qu'il ne les utilise pas [45]. Les cookies sont des fichiers texte de petite taille dans lesquels sont stockées des informations légitimes telles que le contenu d'un panier sur un site de commerce électronique, des informations permettant de gérer la session d'un utilisateur ou encore de sauvegarder des préférences pour un site internet comme la langue utilisée par l'utilisateur. Cependant, les cookies peuvent jouer d'autres rôles tels que le traçage d'utilisateurs lors de leurs navigations internet ou l'affichage de publicité. Par exemple, le cookie Doubleclick permet de savoir si une publicité a déjà été affichée chez un utilisateur pour éviter de la réafficher. Google, propriétaire de DoubleClick, précise que « [...] aucune information permettant d'identifier personnellement l'utilisateur n'est enregistrée dans le cookie par DoubleClick » [19]. Cependant, même sans le

18. En regardant la liste des catégories des centres d'intérêt en 2015, on peut voir qu'il y en a environ 2000 possibles - `support.google.com/ads/answer/2842480?hl=fr&ref_topic=2971788&vid=0-635761091585781030-3312563870` - accessible 01-09-2015

19. `support.google.com/adsense/answer/2839090?hl=fr` - accessible le 01-09-2015

stockage d'informations permettant d'identifier un utilisateur personnellement, il est techniquement possible de le tracer sur tous les sites internet faisant appel au service DoubleClick ce qui peut amener à une désanonymisation.

2.2.2. Attaques par inférence

L'objectif d'une attaque par inférence est de déduire des informations sur un individu à l'aide potentiellement d'informations auxiliaires. Par exemple, ce type d'attaque a été menée contre la base de données de Netflix en utilisant celle-ci ainsi que celle de IMDB (exemple 4 ci-dessous) [94]. On peut aussi prendre comme exemple une attaque par inférence sur des données de mobilité. Supposons par exemple que l'adversaire ait pu déduire les coordonnées (lattitude, longitude) suivantes (48.125068, -1.623833999999988) et (48.110886, -1.655885). La première paire de coordonnées représente l'endroit le plus visité pendant la journée alors que la deuxième paire de coordonnées est l'endroit le plus visité pendant la nuit. En interrogeant Google Maps on peut apprendre que la première coordonnée correspond à Supelec, il doit donc s'agir de son lieu de travail. On peut donc déduire que la deuxième paire de coordonnées correspond à son domicile. Une fois l'adresse de la personne identifiée, il est possible de retrouver son nom à l'aide d'un simple annuaire.

Il existe plusieurs types d'attaques par inférence que nous survolerons ci-après.

Attaques par chaînage Le but de l'adversaire est de relier des enregistrements stockés dans des bases de données différentes contenant une fraction de personnes en commun [123]. Ce type d'attaque correspond à l'exemple d'AOL qui est évoqué ci-dessous (exemple 3). Les conséquences de cette attaque ont été désastreuses pour AOL. Une attaque par chaînage peut être considérée comme une forme spécifique d'attaque par désanonymisation.

Attaques par composition Le principe de l'attaque par composition (aussi appelée attaque par intersection) est d'inférer des informations sur une personne en croisant deux bases de données assainies. L'exemple 6 présente ci-après une attaque par composition. Il s'agit d'un exemple minimaliste dont le but pédagogique permet de montrer le fonctionnement d'une telle attaque.

2.3. Mécanismes de protection de la vie privée

Nous présentons dans cette section quelques mécanismes de protection de la vie privée. Tout d'abord, nous survolerons des niveaux de respect de la vie privée existants (anonymat, pseudonymat) dans la sous-section 2.3.1, avant de présenter l'assainissement de données dans la section 2.3.2. Pour finir, nous présenterons quelques primitives développées en cryptographie pour la protection de la vie privée (sous-section 2.3.3).

2.3.1. Techniques pour protéger la vie privée

Pour se protéger des attaques précédentes, il est important d'utiliser des technologies spécifiques (PIR, canal de communications anonymes, etc., présentées dans cette section) ainsi que des algorithmes et méthodes cryptographiques (chiffrement homomorphe, schéma de signature de groupe, etc.). Tout d'abord, il faut savoir qu'en protection de la vie privée, il existe quatre différents niveaux de protection :

1. *Pseudonymat :* le but du pseudonymat est d'utiliser un pseudonyme sur internet (par exemple, forum, chat, ...) à la place de son vrai nom. Cette technique est jugée non sûre depuis Sweeney [122] ainsi que dans les affaires AOL et Netflix (voir l'exemple 3).
2. *Anonymat :* l'anonymat est le fait de pouvoir effectuer une action (par exemple, achat en ligne, paiement en magasin, ...) sans que cette action soit liée à une identité particulière.
3. *Non-chaînabilité :* le but est de ne pas pouvoir relier au moins deux actions qui ont été réalisées de manière anonyme (par exemple, achat en liquide de billet de bus et courses réalisées en magasin).
4. *Non-observabilité :* la non-observabilité est le fait de ne pas savoir quelles actions ont été réalisées.

Il faut noter que non-observabilité ⇒ non-chaînabilité ⇒ anonymat ⇒ pseudonymat, avec « ⇒ » exprimant l'implication.

Exemple 3. *Affaire AOL. Le pseudonymat est jugé insuffisant depuis l'affaire d'AOL. En effet, AOL a publié les requêtes effectuées par des utilisateurs sur son moteur de recherche en 2006 en mettant un pseudonyme à la place du nom de l'utilisateur. Cependant, certaines personnes ont réussi, à l'aide d'attaques par inférence, à retrouver le nom de certains utilisateurs, et donc à briser la « protection » mise en œuvre pour protéger leur vie privée.* [20]

Exemple 4. *Netflix Prize* [21]*. Le* Netflix Prize *fut une compétition proposée par Netflix pour trouver un nouvel algorithme de recommandation prédisant les notes des utilisateurs en utilisant celles fournies par le passé sans utiliser d'autres informations. Pour cela, Netflix donna une base de données de test contenant 100 480 507 notes données par 480 189 utilisateurs sur 17 770 films. Dans la base de données, les utilisateurs n'apparaissaient que sous forme d'entiers, pour protéger leur vie privée, tout comme le nom des films. Cependant, deux chercheurs de l'université du Texas ont réussi en 2007 à identifier des utilisateurs en réalisant une attaque par chaînage (Définition 2.2.2 Section 2.2).*

20. `http://www.nytimes.com/2006/08/09/technology/09aol.html?pagewanted=1&_r=1` – Accès le 07 mars 2016
21. `en.wikipedia.org/wiki/Netflix_Prize` – Accès le 16 février 2016

Pour protéger la vie privée des utilisateurs, différentes techniques supplémentaires existent, telles que la perturbation des informations, l'assainissement de données ou encore l'utilisation de techniques cryptographiques sûres. Ces différentes techniques sont explicitées si dessous.

2.3.2. L'assainissement de données

L'assainissement de données est un processus qui permet, dans certains contenus (par exemple, documents confidentiels) de supprimer n'importe quelles informations sensibles avant leur diffusion. Un but pratique de l'assainissement de données est la déclassification de documents confidentiels, ou encore la diminution de leur classification. Dans une base de données, il faut par exemple utiliser des généralisations, des suppressions, etc. Le problème qui se pose est le suivant : quel est le compromis qui doit être réalisé entre les modifications liées à l'assainissement (certaines données doivent rester utilisables, par exemple, statistiques) et le niveau de protection de vie privée souhaité.

Les techniques utilisées pour assainir les données numériques sont : perturbation, agrégation, généralisation et suppression. C'est notamment ce que fait le *k-anonymat* (*k-anonymity* en anglais)[122]. Pour pallier certaines faiblesses de cette dernière, Machanavajjhala, Kifer, Gehrke et Venkitasubramaniam ont proposé un nouvel algorithme appelé *ℓ-diversity* en 2007 [85]. Il existe aussi la *confidentialité différentielle* ou *Differential Privacy* proposée par Dwork en 2006 [50].

_k_-anonymat Le but du *k-anonymat* est d'assainir les informations relatives à des utilisateurs en utilisant des généralisations et des suppressions qui impliquent la création de groupes de k individus. Dans chaque groupe d'individus formé, les attributs sensibles d'un tuple appartenant à un individu ne peuvent pas être distingués des attributs sensibles des $k-1$ individus présents. De plus, les identifiants sont supprimés et les quasi identifiants sont généralisés par l'algorithme de k-anonymat.

En 1998, Sweeney a proposé *Datafly*, un système préservant l'anonymat de données médicales [117]. Pour ce faire, le système utilise des généralisations, des suppressions, des substitutions et des insertions dans la base de données. Par exemple, les quasi-identifiants tels que les numéros de sécurité sociale seront automatiquement fabriqués, ou encore la date d'anniversaire sera généralisée à l'année. *Datafly* utilise des heuristiques dans le but de faire des approximations, mais celles-ci ne sont pas optimales. En termes plus généraux, les identifiants sont fabriqués et les quasi identifiants sont généralisés. Par exemple, un numéro de sécurité sociale sera supprimé et un numéro de remplacement sera entièrement fabriqué. La généralisation de quasi-identifiant entraine une perte d'information. Par exemple, un code postal 35760 peut, suivant la généralisation effectuée, devenir 35***. L'information apportée n'est donc plus la ville, mais le département dans lequel l'utilisateur vit.

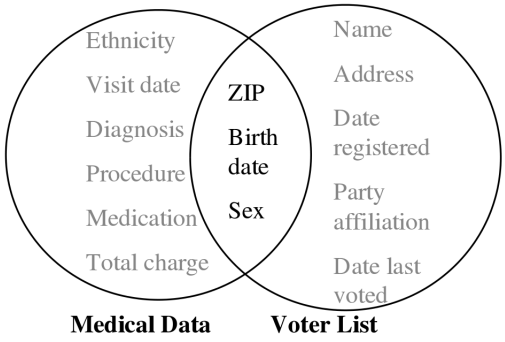

FIGURE 2.1.: Croisement de bases de données ayants des attributs en commun. [123]

Sweeney a montré en 2000 [122] qu'il était possible de désanonymiser 87% des Américains en utilisant des quasi-identifiants (sexe, code postal, date de naissance) et des informations auxiliaires. La désanonymisation d'une base de données médicales a été effectuée à l'aide de la liste électorale de Cambridge Massachussetts (Figure 2.1). Il est à noter qu'un ensemble de quasi identifiants peut être un identifiant. En 2002, Sweeney propose dans [123] un algorithme théorique nommé *Preferred Minimal Generalization* permettant de faire des suppressions et des généralisations pour rendre les données anonymes tout en essayant de conserver l'utilisabilité des données.

L'exemple 5 suivant montre le résultat d'une exécution d'un algorithme de 2-anonymat et de 3-anonymat. Comme on peut le voir pour le tableau contenant le résultat de l'algorithme de 3-anonymat, les données sont plus dégradées qu'avec l'algorithme 2-anonymat. Il faut donc faire attention à protéger la vie privée des utilisateurs présents dans la base de données tout en conservant un niveau d'utilisabilité correcte.

Exemple 5. *k-anonymat.Considérons que deux individus récupèrent, pour des statistiques, une base de données anonymisée avec du* k*-anonymat auprès d'un hôpital. Cette base de données contient des informations personnelles et sensibles telles que le nom du patient, le code postal, l'âge, le sexe et les maladies. Les données brutes sont présentées dans le tableau 2.1. Tandis que les données après le passage de celles-ci dans l'algorithme 2-anonymat sont présentées dans le tableau 2.2, et celles avec le 3-anonymat dans le tableau 2.3. Une fois assainies, les données telles que le code postal ne sont pas connues entièrement. La seule information que cela offre est le département ou une liste de départements de résidence possibles. Les âges sont contenus dans un intervalle et leur genre est supprimé. De plus, la maladie est la seule information exacte connue.*

Exemple 6. *Attaque par composition détaillée. Les tableaux 2.4 et 2.5 contiennent des données de deux hôpitaux différents. Ces données ont été anonymisées à l'aide d'une méthode de* k*-anonymat et chacun des tableaux contient des données sensibles concernant les antécédents médicaux de patients. Supposons que nous savons que Alice est âgée de 20 ans, qu'elle habite dans le département Ille-et-Vilaine (35) et qu'elle soit patiente des deux hôpitaux. On peut se poser la question concernant ce qu'il est possible*

Nom	Code postal	âge	Sexe	Maladie
Dupont	35000	26	F	Cancer
Henry	35045	30	H	Grippe
Laplace	35115	45	F	Diabète
Durant	36050	52	F	Cancer
Wayne	36150	18	H	Asthme
Néper	36170	19	H	Diabète

Tableau 2.1.: Données brutes provenant d'un hôpital. Les données doivent être anonymisées pour protéger la vie privée des patients, avant de distribuer la base de données.

Nom	Code postal	âge	Sexe	Maladie
*	35***	$18 \leq x \leq 25$	*	Cancer
*	35***	$18 \leq x \leq 25$	*	Grippe
*	35***	$25 < x < 35$	*	Diabète
*	35***	$25 < x < 35$	*	Cancer
*	36***	$50 > x > 40$	*	Asthme
*	36***	$50 > x > 40$	*	Diabète

Tableau 2.2.: Données après le passage de celles-ci dans l'algorithme 2-anonymat

Nom	Code postal	âge	Sexe	Maladie
*	3****	$18 \leq x < 30$	*	Cancer
*	3****	$18 \leq x < 30$	*	Asthme
*	3****	$18 \leq x < 30$	*	Diabète
*	35***	$30 \leq x < 55$	*	Grippe
*	35***	$30 \leq x < 55$	*	Diabète
*	35***	$30 \leq x < 55$	*	Cancer

Tableau 2.3.: Données après le passage de celles-ci dans l'algorithme 3-anonymat.

Nom	Code postal	Âge	Sexe	Maladie
*	35***	≤ 25	*	Cancer
*	35***	≤ 25	*	Grippe
*	35***	< 35	*	Diabète
*	35***	< 35	*	Cancer
*	36***	> 35	*	Asthme
*	36***	> 35	*	Diabète

Tableau 2.4.: Base de données 1

Nom	Code postal	Âge	Sexe	Maladie
*	36***	≥ 30	*	Parkinson
*	36***	≥ 30	*	Asthme
*	36***	≥ 30	*	Infection respiratoire
*	35***	< 30	*	Alzheimer
*	35***	< 30	*	SIDA
*	35***	< 30	*	Cancer

Tableau 2.5.: Base de données 2

d'apprendre à partir de ces connaissances auxiliaires. Tout d'abord, comme Alice habite dans le 35 on peut se concentrer sur les lignes des tableaux correspondants à cette information. De plus, comme Alice à 20 ans on peut se focaliser sur deux premières lignes du tableau 2.4 car l'âge des individus associés à ces lignes est inférieur à 25 ans et les trois dernières lignes du tableau 2.5 car leur âge y est inférieur à 30 ans. La seule maladie que ces lignes ont en commun est le cancer.

Cependant, Machanavajjhala, Kifer, Gehrke et Venkitasubramaniam ont montré en 2007 [85] que le k-anonymat est sensible aux attaques par connaissance auxiliaire (qui peuvent être rapprochées des attaques par chaînage) ainsi qu'à celles par homogénéité. La faiblesse appelée homogénéité vient du fait qu'une fois les tuples anonymisés dans un groupe, il ne faut pas que tous les individus de ce groupe aient la même valeur sensible. En effet, si l'adversaire possède les quasi identifiants, il sera sûr que la personne qu'il recherche a cette valeur sensible précise. De plus, en reliant deux bases de données anonymisées avec le k-anonymat, l'apprentissage de valeurs précises est possible, ce qui brise l'anonymat. Pour contrer ce type de menace, la recherche s'est orientée vers la *l-diversité*.

Confidentialité différentielle La *confidentialité différentielle* a été proposée par Cynthia Dwork en 2006 [50]. Cynthia Dwork a aussi proposé en 2008 une étude sur les résultats de la confidentialité différentielle [51]. La confidentialité différentielle a pour but de se protéger, rigoureusement et statistiquement, contre ce qu'un adversaire peut inférer de la connaissance des résultats. En d'autres termes, un individu n'apprendra aucune information spécifique à propos d'un autre, mais il pourra apprendre des informations utiles sur un groupe d'individus. Par exemple, il peut être utile de connaître l'âge

moyen d'un groupe d'individus sans pouvoir apprendre l'âge d'une personne spécifique. De plus, la confidentialité différentielle permet de faire en sorte qu'un enregistrement, qu'il soit présent ou non dans la base de données, n'entraîne aucune fuite d'information. Il s'agit donc d'un mécanisme d'analyse de données préservant la vie privée qui ne constitue pas, contrairement au k-anonymat, une technique de protection de la vie privée. Elle permet de contrer les risques de révéler de l'information lors de la jointure de deux bases de données statistiques. La confidentialité différentielle peut être instanciée par différentes techniques comme la perturbation de données. Le problème majeur de la confidentialité différentielle est le fait que les données peuvent ne plus être utilisables après perturbation.

2.3.3. La cryptographie pour la protection de la vie privée

Réseaux de communications anonymes Un *canal de communication anonyme*, introduit par Chaum [35] en 81, permet à un utilisateur d'envoyer un message à un autre utilisateur de telle sorte que les identités de l'expéditeur et du destinataire demeurent toutes deux cachées. Par exemple, TOR (*The Onion Routing*) [47] est un réseau de communication anonyme utilisé quotidiennement par plus de 600 000 personnes. L'un des atouts de TOR est que la préservation de la vie privée des utilisateurs ne repose pas seulement sur une seule entité, mais plutôt sur la confiance qui est répartie entre plusieurs nœuds du réseau TOR.

Le réseau de mélange (MixNet en anglais) [35] ainsi que les DC-net (Dîner des cryptographes) [36], tous deux introduits par Chaum respectivement en 1981 et 1988, sont d'autres exemples de réseaux célèbres de communications anonymes. En 1981, Chaum [35] a introduit le concept de *MixNet*. Un *MixNet* est composé de routeurs (*mix*) qui permettent de cacher le lien entre les messages entrants et sortants. Ces routeurs prennent les messages en entrée, les déchiffrent, attendent un certain temps (par exemple, il attend 1000 messages) et permute les messages avant de les envoyer. Cette concaténation successive de routeurs rend plus difficile la reconstitution des liens entre messages entrants et sortants.

Dîner des cryptographes Le problème du dîner des cryptographes fût introduit par David Chaum [36] :

Trois cryptographes dînent dans un restaurant. Le serveur les informe qu'il est possible de payer de manière anonyme. C'est-à-dire que seul le payeur saura qui a réglé l'addition. Cependant, les trois cryptographes veulent savoir si c'est bien l'un d'eux qui a payé et non la NSA.

Pour résoudre ce problème, les trois cryptographes agissent comme suit :

1. Tous les cryptographes lancent une pièce (non biaisée) de telle sorte que seuls le lanceur et le cryptographe à sa droite voient le résultat.

2. Tous les cryptographes annoncent à voix haute si les deux pièces (celle du cryptographe à sa gauche et la sienne) sont sur la même face ou non. Cependant, le payeur annonce le contraire de ce qu'il voit. Un nombre pair de différences indique que la NSA a payé, sinon c'est l'un des cryptographes, mais personne ne sait lequel.

Ce protocole peut être généralisé pour un nombre quelconque de cryptographes. De plus, c'est un protocole qui permet d'envoyer des messages anonymement, et lisibles, uniquement par le receveur légitime s'ils sont chiffrés [12].

Accréditations anonymes Les *accréditations anonymes* pourraient être utilisées tous les jours (passeport, permis de conduire, carte d'accès,...). Une *accréditation* est une autorisation d'accès à une ressource pour un individu donné. L'accréditation est fournie par une autorité spécifique et celle-ci est reliée à une identité. Elle contient des données à caractère personnel (nom, adresse, biométrie, etc). C'est pourquoi les chercheurs se sont intéressés aux accréditations anonymes. Les accréditations anonymes permettent l'autorisation d'accès à une ou plusieurs ressources sans dévoiler l'identité de l'individu qui y accède ni ses informations personnelles. L'anonymat et potentiellement la non-chaînabilité garantissent le respect la vie privée des utilisateurs. Il y a deux types d'usages de ces accréditations : *unique* et *multiple*. L'usage unique permet de n'utiliser qu'une seule fois l'accréditation (monnaie électronique, etc), tandis que l'usage multiple permet d'utiliser la même accréditation plusieurs fois (carte de transport anonyme, etc). Cependant, le dernier usage permet de lier différentes actions à un même individu même si son identité n'est pas dévoilée.

Un type d'accréditation anonyme est la *signature de groupe*. Les *signatures de groupe* ont été introduites par Chaum et van Heyst [37] pour garantir qu'un message a été envoyé par un membre légitime d'un groupe, sans révéler l'identité du signataire. Au moment de l'enregistrement, chaque membre reçoit une clef privée de l'*autorité d'enregistrement*. Le message signé avec une clef privée légitime est vérifiable en utilisant la clef publique du groupe. De plus, les signatures de groupe sont non-chaînables dans le sens où un adversaire ne peut pas savoir si deux messages ont été signés avec la même clef privée donc, par extension, si c'est la même personne qui a produit la signature. Néanmoins, une entité différente, appelée *autorité d'ouverture*, peut identifier quel membre du groupe a signé un message spécifique, dans le but de révoquer l'anonymat de ce membre. Cette procédure est réalisée seulement si un signataire particulier est malveillant. Enfin, une *autorité de révocation* (qui peut être indépendante ou non de l'autorité d'enregistrement et d'ouverture) peut aussi révoquer les membres d'un groupe.

Une signature de groupe est un tuple des six algorithmes suivant : $\texttt{GSScheme} = (\texttt{GSKGen}, \texttt{Join}, \texttt{Sign}, \texttt{Vf}, \texttt{Open}, \texttt{Revoke})$.

— L'algorithme `GSKGen` permet de générer les clefs du manager du groupe.
— Le protocole `Join` permet à un utilisateur de s'enregistrer auprès du groupe manager et d'obtenir une clef de signature de groupe.
— L'algorithme `Sign` est utilisé par un membre du groupe pour signer un message.

— L'algorithme `Vf` permet de vérifier la validité d'une signature.
— L'algorithme `Open` permet d'ouvrir une signature pour révéler l'identité du signataire.
— L'algorithme `Revoke` est utilisé pour invalider la clef de signature de groupe de l'un des membres.

Les *preuves à divulgation nulle de connaissance* peuvent aussi être utilisées en ce sens. Les *preuves à divulgation nulle de connaissance* (ZK-PK), introduites par Goldwasser, Micali et Rackoff [60] en 1985, permettent à une partie appelée *prouveur* de prouver qu'un état est vrai à un *vérifieur* de telle sorte qu'aucune information additionnelle autre que la véracité ne soit connue. Une ZK-PK peut aussi être non interactive (NIZP-PK). Les *preuves à divulgation nulle de connaissance non interactives*, présentées par Blum, Feldman et Micali [20], permettent au *prouveur* de prouver qu'il connaît un témoin spécifique d'un état publiquement connu à un *vérifieur*, de telle sorte qu'aucune information additionnelle autre que la véracité ne soit connue et le protocole se déroule de manière non interactive (asynchrone). Ainsi la preuve est vérifiée après sa publication, potentiellement longtemps après. Nous appelons le processus de création de preuves réalisé par le prouveur `ProofGen` et la phase de vérification réalisée par le vérifieur d'une manière non interactive `ProofVerif`. Groth, Ostrovsky et Sahai [65] ont montré comment construire une NIZK-PK pour un langage NP arbitraire. Une ZK-PK peut avoir deux buts, identifiés ci-dessous :

— *Preuve d'appartenance à un ensemble :* cette preuve permet au prouveur de convaincre le vérifieur que son témoin appartient à un ensemble fini connu par le vérifieur.
— *Preuve de conformité :* cette preuve permet au prouveur de convaincre de la conformité d'une opération.

Récupération de données Le *retrait d'informations privé* (*Private Information Retreival* ou *PIR* en anglais) [99, 44, 84, 40, 59] permet de récupérer une information dans une base de données publique de telle sorte que le propriétaire de la base de données ne sache pas de quelle information il s'agit. La solution la plus simple est de récupérer toute la base de données pour n'extraire que l'information utile. Cependant, pour de grandes bases de données, il n'est pas possible de procéder ainsi, car cela pose un problème de bande passante. Plutôt que de demander toute la base de données, le client envoie une requête qui sera exécutée par le serveur. Ensuite, le serveur envoie le résultat de la requête au client qui pourra la décoder. Une manière de réaliser un *PIR* est d'utiliser le chiffrement homomorphe (expliqué ci-dessous).

Les protocoles de *Transfert Équivoque* (*Oblivious Transfer* ou *OT* en anglais) constituent un sous-ensemble des techniques de retrait d'informations privé. Le Transfert Équivoque (appelé par la suite *OT* pour respecter la littérature) est une primitive cryptographique (Fig. 2.2) qui permet à une partie appelée *émetteur*, d'envoyer une valeur parmi plusieurs à un *receveur*, de telle sorte que : 1) l'émetteur ne sait pas quelle valeur est récupérée par le receveur et 2) le receveur n'apprend aucune information sur les autres valeurs qu'il n'a pas piochées. Cette primitive a été originalement introduite par Rabin [110] et par la suite étudiée et améliorée [93, 21, 41, 26, 61, 62, 83].

La différence entre un *PIR* et un OT réside dans le fait que les informations fournies en paramètres de l'OT sont privées. Les implémentations d'OT existent pour les adversaires *honnête-mais-curieux*, qui suivent correctement le protocole, mais essaient d'apprendre autant d'informations que possible sur les entrées des différentes parties, et pour les adversaires *malveillants* qui essaient de tricher arbitrairement. Dans cette contribution, nous adoptons les définitions de sécurité introduite par Noar et Pinkas dans le modèle « half-simulation ».

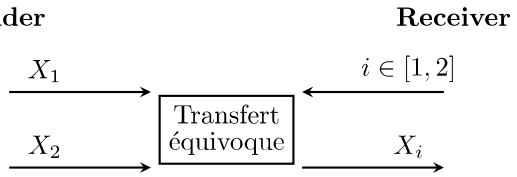

FIGURE 2.2.: Transfert équivoque 1 parmi 2

Chiffrement homomorphe Le *chiffrement homomorphe* [115, 100, 58, 58, 14, 52, 48] est différent des chiffrements symétrique et asymétrique. Pour un état de l'art détaillé sur le chiffrement homomorphe, nous renvoyons le lecteur à Guellier [67]. En effet, le chiffrement homomorphe permet de faire des calculs (addition et multiplication) directement sur les chiffrés tout en conservant l'exactitude du résultat. Le résultat déchiffré est le même que celui qui aurait été obtenu si le calcul avait été effectué sur les données en clair. Formellement, considérons que $\forall m_1, m_2 \in M$ avec M l'espace des messages possibles et que *Enc* est la fonction de chiffrement avec une clef *key* il existe la relation

$$Dec(Enc(m_1) \odot_1 Enc(m_2)) = m_1 \odot_2 m_2 \tag{2.1}$$

avec $\odot$ un opérateur additif ou multiplicatif. Un *chiffrement homomorphe complet* est un chiffrement homomorphe qui est simultanément additif et multiplicatif. Cependant, le coût est élevé, mais tend à s'améliorer. Il est par exemple utilisé pour faire du Retrait d'Informations Privé (Private Information Retreival ou PIR en anglais) ou du tatouage de document. Le PIR permet de récupérer une information dans une base de données sans que le propriétaire sache ce qui intéresse le receveur. *Un utilisateur veut récupérer une information dans une base de données sans que le propriétaire de la base de données sache ce qu'il a pris.* Considérons une base de données de la forme :

1	2	3	4	5	6
4	5	2	1	9	3
5	9	2	3	7	1

L'utilisateur va chiffrer chaque élément d'un vecteur de façon indépendante avec sa clef publique. Le vecteur ne va contenir que des chiffrés de *0* et un chiffré de *1* à l'indice de la donnée qu'il veut récupérer, ici une ligne. Le vecteur sera donc de la forme $[Enc_{pk_U}(0), Enc_{pk_U}(1), Enc_{pk_U}(0)]$ pour récupérer la deuxième ligne. Après avoir chiffré chaque élément dans ce vecteur, il va l'envoyer au serveur gérant la base de

données ainsi que sa clef publique. La base de données va chiffrer chaque élément et effectuer la multiplication de chaque élément du vecteur avec chaque élément de chaque colonne avant de faire une addition et de mettre le résultat dans un nouveau vecteur qu'il renverra à l'utilisateur. Cela va donner pour cet exemple :

1. $Enc_{pk_U}(0).Enc_{pk_U}(1) + Enc_{pk_U}(1).Enc_{pk_U}(4) + Enc_{pk_U}(0).Enc_{pk_U}(5) = Enc_{pk_U}(4)$
2. $Enc_{pk_U}(0).Enc_{pk_U}(2) + Enc_{pk_U}(1).Enc_{pk_U}(5) + Enc_{pk_U}(0).Enc_{pk_U}(9) = Enc_{pk_U}(5)$
3. $Enc_{pk_U}(0).Enc_{pk_U}(3) + Enc_{pk_U}(1).Enc_{pk_U}(2) + Enc_{pk_U}(0).Enc_{pk_U}(2) = Enc_{pk_U}(2)$
4. $Enc_{pk_U}(0).Enc_{pk_U}(4) + Enc_{pk_U}(1).Enc_{pk_U}(1) + Enc_{pk_U}(0).Enc_{pk_U}(3) = Enc_{pk_U}(1)$
5. $Enc_{pk_U}(0).Enc_{pk_U}(5) + Enc_{pk_U}(1).Enc_{pk_U}(9) + Enc_{pk_U}(0).Enc_{pk_U}(7) = Enc_{pk_U}(9)$
6. $Enc_{pk_U}(0).Enc_{pk_U}(6) + Enc_{pk_U}(1).Enc_{pk_U}(3) + Enc_{pk_U}(0).Enc_{pk_U}(1) = Enc_{pk_U}(3)$

Le vecteur final contiendra donc : [$Enc_{pk_U}(4)$, $Enc_{pk_U}(5)$, $Enc_{pk_U}(2)$, $Enc_{pk_U}(1)$, $Enc_{pk_U}(9)$, $Enc_{pk_U}(3)$]. Une fois que l'utilisateur le recevra il pourra le déchiffrer avec sa clef privée et il obtiendra bien la deuxième ligne de la base de données : [4,5,2,1,9,3]. Le propriétaire de la base de données ne pourra pas savoir ce qu'il a récupéré et l'utilisateur n'aura aucune information sur les autres données contenues dans la base de données. Cependant, pour que le propriétaire ne sache pas ce que l'utilisateur récupère, les chiffrés d'un même élément doivent être différents. Ce type de chiffrement est appelé probabiliste. Par exemple, chiffrer deux *0* avec la même clef publique doit donner des chiffrés différents et peut donner 012945436518 pour le premier et 641213545431 pour le deuxième. En effet, vu que le serveur contenant la base de données connaît la clef publique de l'utilisateur il ne faut pas qu'il puisse chiffrer, par exemple, un *1* et le comparer aux éléments contenus dans le vecteur sinon il pourrait déduire où se situe le *1* et dans ce cas le retrait d'informations ne serait plus privé.

2.4. L'anonymat et les protocoles de personnalisation de contenus numériques

2.4.1. But, propriétés et entités

L'objectif des protocoles de personnalisation de contenu avec respect de la vie privée est de garantir la possibilité pour l'utilisateur de garder privés ses habitudes, ses goûts, son style de vie, etc. En d'autres termes, le vendeur ne doit pas pouvoir établir de profils sur les acheteurs qui ne souhaitent pas être fichés. Pour cela, le vendeur ne doit pas pouvoir identifier un tel acheteur ni savoir si deux ventes sont réalisées pour ce même acheteur. Ces protocoles sont des améliorations de protocoles asymétriques dans le sens où ils prennent en compte la protection de la vie privée des acheteurs. Un

protocole de personnalisation anonyme doit respecter, en sus des propriétés de sécurité des protocoles de personnalisation de contenus asymétriques [22], les propriétés de respect de vie privée suivantes :

1. *Non-chaînabilité des acheteurs :* le vendeur ne doit pas pouvoir faire le lien entre l'acheteur et le contenu acheté. Cette propriété garantit que le vendeur ne peut pas savoir si deux contenus correspondant à deux transactions différentes sont reliés au même acheteur.
2. *Anonymat révocable :* le vendeur ne doit pas connaître l'identité de l'acheteur ni aucune information permettant de l'identifier directement ou indirectement (nom, prénom, adresse IP, ...) tant que ce dernier n'agit pas de manière malhonnête. L'anonymat doit donc pouvoir être levé, mais uniquement sur une preuve de la culpabilité de l'acheteur suspect.

Dans le cas où le protocole de personnalisation de contenu est monétaire, alors il doit en plus respecter la propriété suivante :

— *Paiement anonyme :* lorsque l'acheteur paie le document, la transaction financière doit être anonyme. Cette propriété n'est pas obligatoire. En effet, un protocole de personnalisation de contenu n'est pas obligatoirement monétaire. C'est-à-dire que le contenu peut être gratuit.

Pour finir, une propriété intéressante, mais que peu de protocoles atteignent est la propriété de *Non-chaînabilité des contenus* ou *Item-Unlinkability* en anglais :

— *Non-chaînabilité des contenus :* le vendeur ne sait pas ce que l'acheteur acquiert. Cette propriété va plus loin que celle de non-chaînabilité des acheteurs. Elle est plus facile à respecter si tous les contenus sont gratuits ou ont le même tarif.

Les entités des protocoles anonymes de personnalisation de contenus sont les mêmes que celles des protocoles asymétriques avec en plus :

— *Autorité d'enregistrement :* l'autorité d'enregistrement permet à un acheteur de s'enregistrer et d'acquérir un couple de clefs privée/publique. Cet enregistrement s'effectue à travers un protocole 2-parties entre l'acheteur et l'autorité.
— *Autorité de révocation :* cette entité permet de révoquer l'anonymat de l'acheteur malhonnête.

2.4.2. Protocoles de personnalisation de contenus respectant la vie privée

En matière de respect de la vie privée, une faiblesse importante des protocoles asymétriques est que les vendeurs connaissent les identités des acheteurs ainsi que ce qu'ils ont acheté. Avec ces informations, il est donc possible de constituer des profils d'utilisateurs et donc de connaître leurs goûts, leurs habitudes, etc. Un *protocole anonyme de personnalisation de contenus* va plus loin en protégeant l'anonymat des acheteurs aussi longtemps qu'ils le souhaitent et qu'ils sont honnêtes (c'est-à-dire, qu'ils ne distribuent pas de copie illégale d'un contenu acquis légalement). Les premiers auteurs à avoir suivi cette voie sont Pfitzmann *et co-auteurs* dans [106, 103, 104]. Depuis,

22. Celles-ci ont été présentées dans la section 1.3.3 du chapitre 1

de nombreux articles ont été écrits sur ce sujet tel que [25] et [108] qui proposent tous deux un protocole basé sur les signatures de groupe. Kuribayashi et Tanaka [76] ont, quant à eux, basé leur protocole sur la propriété additive des schémas de chiffrement homomorphes. Plus récemment Abdul, Gaborit et Carré [12] ou encore Rial, Deng, Bianchi, Piva et Preneel [113] ont proposé de nouvelles solutions. Nous reviendrons un peu plus loin plus en détail sur certains de ces protocoles.

Avant les années 2000, seuls Pfitzmann et Waidner [106], et Pfitzmann et Sadeghi [103] ont travaillé sur les protocoles anonymes de personnalisation de contenus. Ils sont partis d'un schéma symétrique pour ensuite l'améliorer et en faire un protocole asymétrique. Ensuite, ils ont ajouté et utilisé des propriétés et des méthodes permettant de respecter la vie privée des acheteurs en cachant leurs identités et en permettant la non-chaînabilité.

Dans [106], Pfitzmann et Waidner proposent une définition des composants d'un protocole anonyme avant de proposer deux constructions. Dans cette proposition, l'anonymat est garanti par un enregistrement de l'acheteur auprès d'une Autorité de Certification (CA). Il n'y a que lui qui connaît la vraie identité de l'acheteur et il ne sait pas ce que celui-ci achète. Dans leur protocole, quatre entités distinctes sont présentes : un acheteur B_n, un vendeur M, une autorité de certification CA et un Juge J en cas de dispute. Le principe est simple, l'acheteur s'enregistre auprès d'une autorité de certification et récupère une information qui servira lors du protocole 2-parties entre lui et le vendeur pour personnaliser le contenu. Du fait de l'enregistrement et de l'ajout de la propriété d'anonymat, cette solution est différente des versions précédentes [105, 107].

L'article [106] propose deux protocoles impliquant entre 2 et 4 parties à chaque étape (B_n, CA, M, J).

Les phases composant la première construction sont les suivantes :

— `Setup` : CA génère un couple de clefs ($\mathsf{pk_{CA}}$,$\mathsf{sk_{CA}}$) et distribue $\mathsf{pk_{CA}}$.

— `BReg` : B_n s'enregistre auprès d'un CA, par exemple, une banque (Figure 2.3). Pour s'enregistrer, B_n génère un couple de clefs publique/privée d'un schéma de signature ($\mathsf{pk_{B_n}}$,$\mathsf{sk_{B_n}}$). Ensuite, B_n signe sous sa vraie identité ce couple de clefs. B_n obtient de CA un certificat $cert$=$\mathsf{Cert_{sk_{CA}}}(\mathsf{pk_{B_n}})$. Ce certificat est en fait une signature prouvant que CA connaît l'identité réelle de B_n.

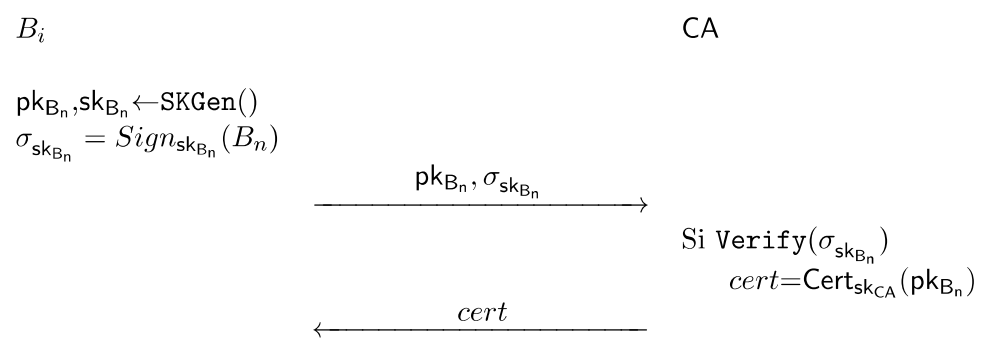

FIGURE 2.3.: Phase d'enregistrement des acheteurs (`BReg`) pour la première construction de Pfitzmann *et co-auteurs* [106]

— `IBuy` : cette phase permet à B_n de récupérer un contenu I_t. B_n signe avec $\mathsf{sk}_{\mathsf{B_n}}$ un texte *text* décrivant la vente ($\sigma^{text}_{\mathsf{sk}_{\mathsf{B_n}}} = Sign_{\mathsf{sk}_{\mathsf{B_n}}}(text)$). L'empreinte f est constituée de $\sigma^{text}_{\mathsf{sk}_{\mathsf{B_n}}}$, *text*, $\mathsf{pk}_{\mathsf{B_n}}$ et *cert*. Puis l'acheteur réalise un engagement sur f, $com = Com(f)$ et envoie ce dernier au vendeur avant de prouver la validité de f et de la certifier de manière nulle de connaissance. La procédure d'insertion n'est pas décrite, ni quels moyens sont utilisés. Cette phase est décrite dans la figure 2.4.

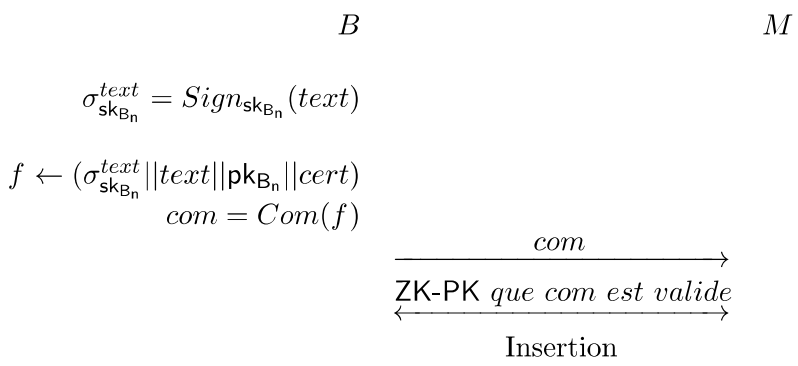

FIGURE 2.4.: Phase d'achat (`IBuy`) de la première construction de Pfitzmann *et co-auteurs* [106]

— `Accuse` : l'identification est un protocole 2-parties entre le vendeur et le CA (Figure 2.9) . Le vendeur commence par extraire f_* d'une copie pirate $\mathsf{FI}^*_{\mathsf{B}^*,\mathsf{I_t}}$. Puis il envoie au CA $\pi_1 = (text, \sigma^{text}_{\mathsf{sk}_{\mathsf{B}_*}}, \mathsf{pk}_{\mathsf{B}_*})$ et demande l'identification de B_*. La preuve que le propriétaire du pseudonyme $\mathsf{pk}_{\mathsf{B}_*}$ a redistribué illégalement le contenu est π_1. Ensuite, CA retourne $\sigma_{\mathsf{sk}_{\mathsf{B}_*}}$ et cette dernière avec π_1 constituent π. Après vérification des valeurs, M engage la procédure d'accusation auprès du juge.
— `ForcedAccuse` : Cette étape intervient si le CA ne veut pas identifier l'acheteur. Dans ce cas, le vendeur prévient le juge J et ils réalisent une identification forcée. M envoie π_1 et *cert* au J pour prouver que CA connaît l'identité de B_*. L'autorité CA doit donc identifier B_* sinon il sera désigné comme coupable.
— `Trial` : le juge vérifie $\sigma_{\mathsf{sk}_{\mathsf{B}_*}}$ et π (Figure 2.6). S'il est convaincu, il accuse B_n sinon il sort $\perp$.

D'après les auteurs, l'autorité de certification n'a pas besoin d'être de confiance, car ils font la supposition forte que si elle est malhonnête elle peut, au pire, seulement refuser une inscription. Les propriétés que les auteurs souhaitent atteindre sont les suivantes :

— *Efficacité :* le protocole doit fonctionner correctement même en cas d'essai de triche avec une sortie normale ou un arrêt du protocole.
— *Intégrité :* la propriété d'intégrité agit auprès de trois entités : vendeur, acheteur et CA. En effet, M doit retrouver au moins un des B ayant distribué un document sans autorisation, et il ne doit pas pouvoir accuser un acheteur innocent. Cette propriété n'est pas atteinte dans la première construction. En effet, la valeur extraite est supposée être une empreinte f valide. Cependant, en cas de collusion,

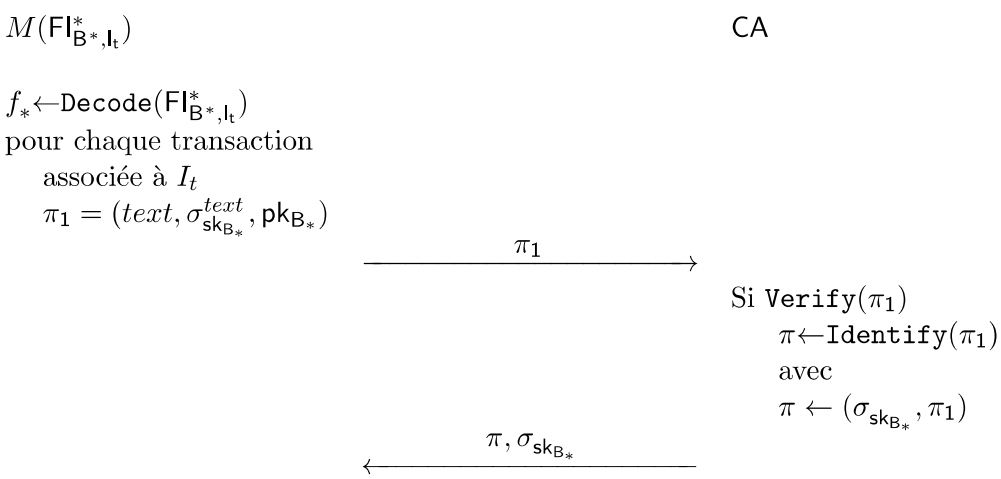

FIGURE 2.5.: Phase d'accusation (`Accuse`) de la première construction de Pfitzmann *et co-auteurs* [106]

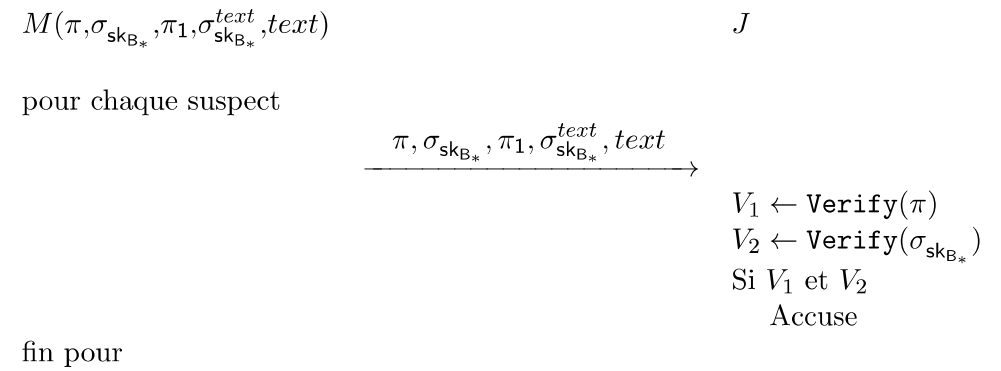

FIGURE 2.6.: Phase de vérification d'accusation (`Trial`) par le juge pour la première construction de Pfitzmann *et co-auteurs* [106]

cette empreinte ne sera pas valide, car elle n'est pas faite pour résister aux collusions. L'intégrité pour B consiste à le protéger des accusations de M tant que B est innocent. Pour finir, CA ne doit pas être marqué comme malhonnête par J alors qu'il ne l'est pas.

— *Anonymat :* aucune information concernant l'acheteur ne doit être dévoilée, sauf dans le cas unique où il serait reconnu coupable de distribution illégale par J.

La deuxième construction s'appuie sur les codes anti-collusion de Boneh et Shaw [22]. Comme on a pu le voir jusqu'à présent, les codes anti-collusion de Boneh et Shaw ont été souvent employés, car ces derniers étaient considérés comme les meilleurs avant 2003. Dans cette deuxième construction, Pfitzmann et Waidner ajoutent une phase de préparation de contenu `IPrep`.

— `IPrep` : le vendeur choisit les emplacements dans le contenu I qui recevront les

bits du code externe f_m^2 de taille m. Il choisit aussi une permutation Π de taille m sur l'alphabet $\chi = \{1 \ldots, q\}$ avec $|\chi| = q$ (Figure 2.7).

$M(I)$

pour chaque $I_{\in}\{I_1, \ldots, I_k\}$
  Choix des emplacements recevant l'empreinte f_m^2
  Choix d'une permutation Π sur $\chi = \{1 \ldots, q\}$

FIGURE 2.7.: Phase de préparation (`IPrep`) de Pfitzmann *et co-auteurs* [106]

Ensuite, les sous-protocoles `IBuy` et `Accuse` sont modifiés avant d'exécuter la même procédure `Trial` que précédemment.

— `IBuy` : le vendeur donne secrètement en entrée I. Le vendeur commence par générer un code interne qui permettra d'encoder le code externe généré. Pour un acheteur B_n, le vendeur choisit k_1 bits de manière aléatoire et secrètement pour chaque symbole du code externe. Le nombre de symboles est m et les symboles du code externe sont $f_2^{'}$=($f_{2,1}^{'}$, ..., $f_{2,m}^{'}$). Ensuite, l'acheteur génère un code $f_2^{''}$ avant de l'encoder avec un code correcteur d'erreur et d'effacement(*Error and Erasure Correcting Code* ou *EECC* en anglais) en m symboles, chacun composé de k_2 bits. $f_2^{''}$ est ensuite engagé auprès du vendeur. À l'aide d'un protocole 2-parties, $f_2^{'}$ (fourni par le vendeur) et $f_2^{''}$ (fourni par l'acheteur) sont mixés de la manière suivante : $f_2 = (\Pi(f_{2,1}^{'}||f_{2,1}^{''}), \ldots, \Pi(f_{2,m}^{'}||f_{2,m}^{''}))$ où Π est la permutation choisie lors de la phase de préparation. La sortie de ce protocole 2-parties est le code externe f_2 de l'acheteur. Ensuite, chaque symbole du code externe f_2 est encodé avec le code interne f_1 (fourni par le vendeur), de nouveau à l'aide d'un protocole 2-parties sécurisé, qui fournit le mot de code f qui sera inséré dans I. Seul l'acheteur a connaissance du contenu personnalisé. Cette phase est décrite dans la figure 2.8.

— `Accuse` : cette phase (décrite dans la figure 2.9) consiste à retrouver un acheteur ayant redistribué illégalement un contenu FI. Le vendeur utilise la procédure de décodage de f_1 pour identifier les m symboles f_{2*}. Pour cela, il utilise Π^{-1} et sépare f_* en k_1 et k_2 bits. Cette séparation permet de retrouver $f_{2*}^{'}$. Ensuite, le vendeur utilise les enregistrements des ventes, pour I, où au moins $\frac{m}{c}$ symboles de $f_{2*}^{'}$ sont présents. Ensuite, il extrait f_{2*}^{code} de f_{2*} et exclu tout symbole différent entre f_{2*}^{code} et $f_2^{''}$. Le résultat est entré en paramètre de la procédure de décodage du code correcteur et le vendeur espère retrouver f_2^{code}.

Comparée à la première construction, celle-ci est plus efficace contre les collusions d'utilisateurs.

Pfitzmann et Sadeghi [103] ont proposé une instanciation concrète de [106] basée sur une méthode de *jetons*. Le jeton est anonyme, mais contient des informations pouvant être extraites dès que le jeton est utilisé plus d'une fois. Ces informations utilisées par CA permettant de révéler l'identité de l'acheteur B_n. Tout d'abord, un acheteur B_n récupère un jeton auprès de CA durant la phase `BReg`. Ensuite, ce jeton est donné au

B		$M(I)$
		Génère $f_2^{'}$=($f_{2,1}^{'}$, …, $f_{2,m}^{'}$) où chaque symbole possède k_1 bits
Génère $f_2^{''}$ f_2^{code}←EECC$f_2^{''}$ $com = Com(f_2^{code})$		
	com, ZK-PK ⟶	
f_2^{code}→	Protocole 2-partie	←$f_2^{'}$
f_2←	$f_2 = (\Pi(f_{2,1}^{'}\|\|f_{2,1}^{code}), \ldots, \Pi(f_{2,m}^{'}\|\|f_{2,m}^{code}))$	
	fin	
f_2→	Protocole 2-partie	←f_1
I_t←	Insertion	
	fin	

FIGURE 2.8.: Phase d'achat (`IBuy`) de la deuxième construction de Pfitzman *et co-auteurs* [106]

M(FI)

f_{2*}←`Decode`(FI,f_1)
reverse← $\Pi^{-1}(f_{2*})$
$f_{2*}^{'}$← `Split`($reverse$)
Pour chaque transaction t
  Si $\frac{m}{c}$ symboles de $f_{2*}^{'}$ sont présents dans $f_2^{'}$
    f_{2*}^{code}← `Split`($reverse$) en k_1 et k_2 bits par symbole

  f_2^{code}←EECC-`Decode`(f_{2*}^{code})

FIGURE 2.9.: Phase d'accusation (`Accuse`) de la deuxième construction de Pfitzman *et co-auteurs* [106]

vendeur M durant la phase `IBuy`. En premier lieu, le vendeur vérifie la validité du jeton. S'il n'est pas valide, il sort $\perp$, sinon la vente se poursuit. Vérifier la validité permet au vendeur de s'assurer que le jeton est bien formé, en d'autres termes, cela signifie que les informations permettant d'identifier un acheteur malhonnête sont bien présentes. L'insertion se fait de la même manière que dans la phase `IBuy` de [106]. Dans ce protocole, le vendeur doit contacter CA pour connaître l'identité de l'acheteur malhonnête. L'autre inconvénient de cette solution est qu'il est nécessaire pour un acheteur de s'enregistrer à chaque fois qu'il veut effectuer un achat, car il doit acquérir

un nouveau jeton.

Camenisch propose dans [25] de modifier l'identification pour permettre au vendeur de connaître directement l'identité de l'acheteur ainsi que l'enregistrement en utilisant un schéma de signature de groupe. Dans sa solution, lorsqu'un utilisateur veut acheter un contenu I il exécute l'algorithme de signature de groupe pour obtenir un couple de clefs publique/privée ($\mathsf{gpk}_{\mathsf{B_n}}$,$\mathsf{gsk}_{\mathsf{B_n}}$) auprès de CA. Ensuite, l'acheteur exécute le schéma de signature de groupe avec l'autorité de levée d'anonymat OA, qui génère la clef de levée d'anonymat, et l'acheteur obtient alors un nouveau couple de clefs publique/privée ($\mathsf{gpk}_{\mathsf{B_{i2}}}$,$\mathsf{gsk}_{\mathsf{B_{i2}}}$). Ensuite, l'acheteur signe un message contenant l'objet de la transaction avec $\mathsf{gsk}_{\mathsf{B_{i2}}}$ envoie la signature ainsi que $\mathsf{gpk}_{\mathsf{B_{i2}}}$ et $Com(\mathsf{gsk}_{\mathsf{B_{i2}}})$. Le vendeur vérifie la signature puis l'acheteur prouve que $Com(\mathsf{gsk}_{\mathsf{B_{i2}}})$ est valide. Ensuite le protocole d'insertion de [103] est exécuté. Celui-ci est un protocole 2-partie en sortie duquel l'acheteur récupère le contenu personnalisé. Le vendeur fournit ce dernier et l'empreinte $Com(\mathsf{gsk}_{\mathsf{B_{i2}}})$ alors que l'acheteur fournit $\mathsf{gsk}_{\mathsf{B_{i2}}}$ et $\mathsf{gsk}_{\mathsf{B_{i2}}}$. Lorsque le vendeur extrait une empreinte de la copie pirate il exécute le code correcteur d'erreur et d'effacement de [103] et retrouve au moins un membre de la collusion. Ce protocole, tout comme [103], ne nous paraît pas résistant aux collusions si celles-ci dépassent ne serait-ce que quelques acheteurs.

La même année que [103], Josep Domingo-Ferrer propose un schéma de personnalisation de contenu anonyme basé sur un protocole de transfert équivoque d'engagements (*Committed Oblivious Transfer* ou *COT*) [49].

- `IPrep` : pour chaque contenu I de taille b bits, le vendeur prépare deux versions I_i^0 et I_i^1 où le i-ème bloc de I reçoit un bit de l'empreinte ($i = \{1..b\}$). Ces deux versions diffèrent uniquement par la position des bits de l'empreinte. En d'autres termes, pour chaque bit i il existe une version marquée et une non marquée. Pour chaque $i \in b$, le vendeur réalise un engagement de bit homomorphique par l'utilisation de XOR [23] sur I_i^0 et I_i^1 et stocke $com_i^0 = Com(I_i^0)$ et $com_i^1 = Com(I_i^1)$. Ensuite, le vendeur envoie un message msg, contenant une courte description de l'item, signé et daté au CA ainsi qu'une liste de $l < b$ positions de bits marqués.
- `BReg` : lors de la phase d'enregistrement, l'acheteur B_n et CA possèdent chacun un couple de clefs publique/privée ($\mathsf{pk}_{\mathsf{B_n}}$,$\mathsf{sk}_{\mathsf{B_n}}$) (respectivement ($\mathsf{pk}_{\mathsf{CA}}$,$\mathsf{sk}_{\mathsf{CA}}$)) d'un schéma de chiffrement ressemblant à El-Gamal. $\mathsf{pk}_{\mathsf{B_n}}$ est le pseudonyme de B_n et il est choisi en fonction de $\mathsf{sk}_{\mathsf{B_n}}$. Ensuite, B_n prouve qu'il connaît $\mathsf{sk}_{\mathsf{B_n}}$ auprès de CA et ce dernier retourne à B_n $cert = Cert(\mathsf{pk}_{\mathsf{B_n}})$.
- `IBuy` : lors de la phase `IBuy`, pour chaque bit i avec $i = \{1...b\}$ du contenu, les étapes suivantes sont réalisées :
  - Le vendeur mélange les paires (I_i^0,I_i^1) et stocke ($I_i^{(0)}$,$I_i^{(1)}$) dans l'enregistrement de la vente.
  - Le vendeur et l'acheteur exécutent un transfert équivoque d'engagement (*Committement Oblivious Transfer*). Pour cela, le vendeur donne en entrée com_i^0 et com_i^1, correspondant à l'engagement de ses deux secrets $I_i^{(0)}$ et $I_i^{(1)}$. B_n fournit en entrée $Com(a_i)$ où a_i est un bit indiquant quel secret il veut

23. pour deux bits, respectivement a_1 et a_2, $Com(a_1)$ x $Com(a_2)$=$Com(a_1 \oplus a_2)$

apprendre. L'acheteur obtient donc le bit I_i^*, qu'il engage auprès du vendeur (com_i^*). B_n envoie aussi la signature $\sigma_{\mathsf{sk}_{\mathsf{B_n}}} = Sign_{\mathsf{sk}_{\mathsf{B_n}}}(com_i^*)$.

— `Accuse` : le contenu redistribué est nommé FI. Le vendeur vérifie d'abord que le contenu original correspondant au contenu redistribué se trouve dans les enregistrements d'achats. S'il n'y est pas, le protocole est arrêté. Sinon, il récupère dans sa liste d'enregistrement des achats toutes les entrées présentes pour ce contenu. Chaque entrée contient les com_i^* signés ($\sigma_{\mathsf{sk}_{\mathsf{B_n}}}$) par les acheteurs pour ce contenu. Le vendeur envoie une copie datée et signée de FI à CA et à tous les acheteurs ayant acquis ce contenu. Ensuite, le vendeur exécute les étapes suivantes pour chaque acheteur :
  — En utilisant un protocole de pile ou face, le vendeur et l'acheteur se mettent d'accord sur l_1 positions de bits de personnalisation avec $l_1 \leq l < b$. S'il y a moins de l_2 marques, alors le protocole est répété jusqu'à obtenir l_3 marques, avec $l_2 \leq l_3 \leq l_1$, l_2 correspondant au nombre minimum de bits de personnalisation à ouvrir.
  — L'acheteur ouvre tous les engagements sur les bits correspondant aux positions l_1.
  — Si tous les engagements ouverts l_3 correspondent aux valeurs des bits de FI, le vendeur obtient la preuve de la redistribution. Sinon, l'acheteur est déclaré innocent et obtient une nouvelle copie.

  Pour finir, le vendeur envoie les engagements ouverts signés à CA pour identifier l'acheteur malhonnête. La preuve de la redistribution est constituée de l'item FI signé précédemment envoyé à CA, des engagements ouverts et signés ainsi que la liste des positions des marques l envoyée à CA durant la phase `IPrep`. Dans le cas où FI ne collabore pas à l'identification, il peut être marqué coupable.

D'après Domingo-Ferrer, ce protocole permet, d'un point de vue de la sécurité :
— d'établir l'authentification de l'acheteur sans compromettre sa clef privée.
— de préserver l'anonymat de l'acheteur.
— d'apporter une sécurité au vendeur : résistance aux collusions.

Cependant, ce protocole ne peut être efficace que contre un nombre très restreint d'adversaires. De plus, l'envoi de l positions de tatouage au CA implique que le vendeur M doit avoir confiance dans le CA. Bien que celui-ci soit une tierce partie de confiance, il paraît plus intéressant de cacher cette information au CA et de ne la lui révéler que dans le cas où une copie pirate est retrouvée. Aussi, lors de l'OT prenant en entrée des engagements le vendeur peut tricher en mettant deux fois la même entrée pour chaque tour. Cela lui permet de savoir quelle est l'empreinte des acheteurs. Certains OT peuvent vérifier que toutes les entrées sont différentes, mais uniquement dans le cas où elles sont en clair ou que le chiffrement est déterministe (par exemple, AES).

En 2001, Sadeghi [116] analyse la sécurité de [49]. Premièrement, Sadeghi considère que, du moment où le protocole utilise une tierce partie de confiance, celui-ci n'est pas anonyme, mais semi-anonyme, car une collusion entre cette tierce partie de confiance et le vendeur est possible avec comme résultat la levée de l'anonymat des acheteurs. Selon lui, un moyen de résoudre ce problème est d'utiliser k tierces parties de confiance jouant le rôle du CA pour rendre plus difficile toute collusion. Cette solution a cepen-

dant l'effet d'alourdir le protocole. De plus, Sadeghi soulève les inconvénients expliqués dans le paragraphe précédent et y répond. En ce qui concerne la divulgation de la liste l, Sadeghi propose de l'envoyer sous forme d'engagement et de ne révéler les informations d'ouverture des engagements qu'au moment de l'accusation. Pour finir, concernant la redondance d'information en entrée du protocole de transfert équivoque (OT), l'acheteur pourrait demander au moment de l'accusation l'ouverture de certaines parties d'engagements réalisés par le vendeur dans le but de vérifier que les deux versions sont différentes.

Lei, Yu, Tsai et Chan ont proposé en 2004 dans [81] un protocole acheteur-vendeur anonyme dérivé de la solution de Memon et Wong [92] présentée dans le Chapitre 1 dans la section 1.6. Lei *et co-auteurs* proposent un protocole qui permettrait de résoudre le problème de liens entre la marque choisie et une transaction spécifique ou un contenu numérique spécifique. Le problème qui se pose si l'empreinte n'est pas liée à la transaction est que le vendeur peut, une fois une copie pirate trouvée, extraire l'empreinte et l'insérer dans un contenu ayant une plus forte valeur monétaire avant d'accuser un acheteur. De plus, il considère que l'acheteur ne doit communiquer qu'avec le vendeur pour réduire le coût des communications jugé, selon eux, comme prohibitif dans le cas où l'acheteur doit communiquer avec plus d'un tiers.

Dans ce protocole, l'acheteur commence par générer une paire de clefs publique/privée ($\mathsf{pk_B}$,$\mathsf{sk_B}$) et fait certifier la clef publique par CA ($cert_{\mathsf{CA}} = Cert(\mathsf{pk_B})$). Cette phase est décrite dans la figure 2.13.

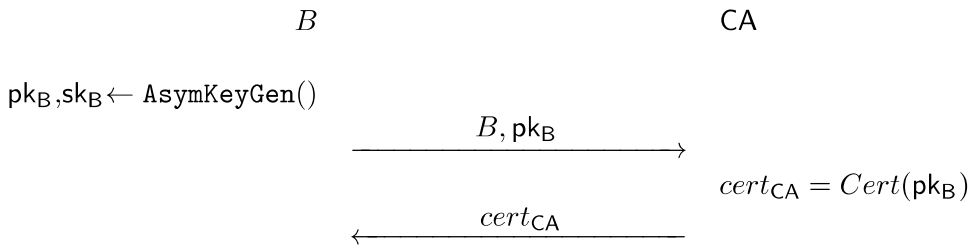

FIGURE 2.10.: Phase d'initialisation de la solution de Lei *et co-auteurs* [81]

Ensuite, l'acheteur et le vendeur négocient un « contrat » (*cont*) qui, entre autres, spécifie le contenu I voulu par l'acheteur. Ensuite, l'acheteur génère une paire de clefs jetable pour un schéma de chiffrement homomorphe ($\mathsf{hpk_B}$,$\mathsf{hsk_B}$).

Puis B génère un certificat sur $\mathsf{hpk_B}$ avec $\mathsf{pk_B}$ ($cert_{\mathsf{pk_B}} = Cert(\mathsf{hpk_B})$). B envoie $cert_{\mathsf{CA}}$, $cert_{\mathsf{pk_B}}$, $\sigma_{\mathsf{sk_{B_n}}} = Sign_{\mathsf{sk_{B_n}}}(text)$ et *cont* à M. Mvérifie la signature et les certificats. Si la vérification est valide, alors Mgénère une empreinte f_1 et l'insère dans le contenu I ($\mathsf{FI} \leftarrow \texttt{Insert}(f_1, I)$). Une fois l'insertion réalisée, le vendeur envoie FI, $cert_{\mathsf{pk_B}}$, $\sigma_{\mathsf{gsk_{B_n}}} = Sign_{\mathsf{sk_{B_n}}}(text)$ et *cont* à une tierce partie de confiance (*WCA* pour *Watermark Certification Authority* ou Autorité de Certification des Empreintes) chargée de générer une empreinte f_2. La tierce partie de confiance chiffre f_2 avec $\mathsf{hpk_B}$ ($Enc_1 = Enc_{\mathsf{hpk_B}}(f_2)$) et aussi avec sa clef publique ($Enc_2 = Enc_{\mathsf{pk_{WCA}}}(f_2)$) qu'elle utilise aussi pour signer $\sigma_{\mathsf{pk_{WCA}}} = Sign(Enc_1, Enc_2, \sigma_{\mathsf{sk_{B_n}}})$ et envoie ces informations

au vendeur. Pour finir, le vendeur insère Enc_1 dans FI et obtient FI' qui se trouve être marqué avec f_1 et f_2. Le vendeur ne connaît que f_1, et n'est donc pas capable de forger un contenu avec l'empreinte d'un acheteur innocent. Le contenu marqué étant chiffré avec $\mathsf{hpk_B}$, seul l'acheteur est capable de le déchiffrer. De plus, le vendeur ne pourra pas utiliser une empreinte dans une copie pirate pour l'insérer dans la copie d'un contenu à plus forte valeur monétaire grâce au *WCA*. Ces étapes sont décrites dans la figure 2.11.

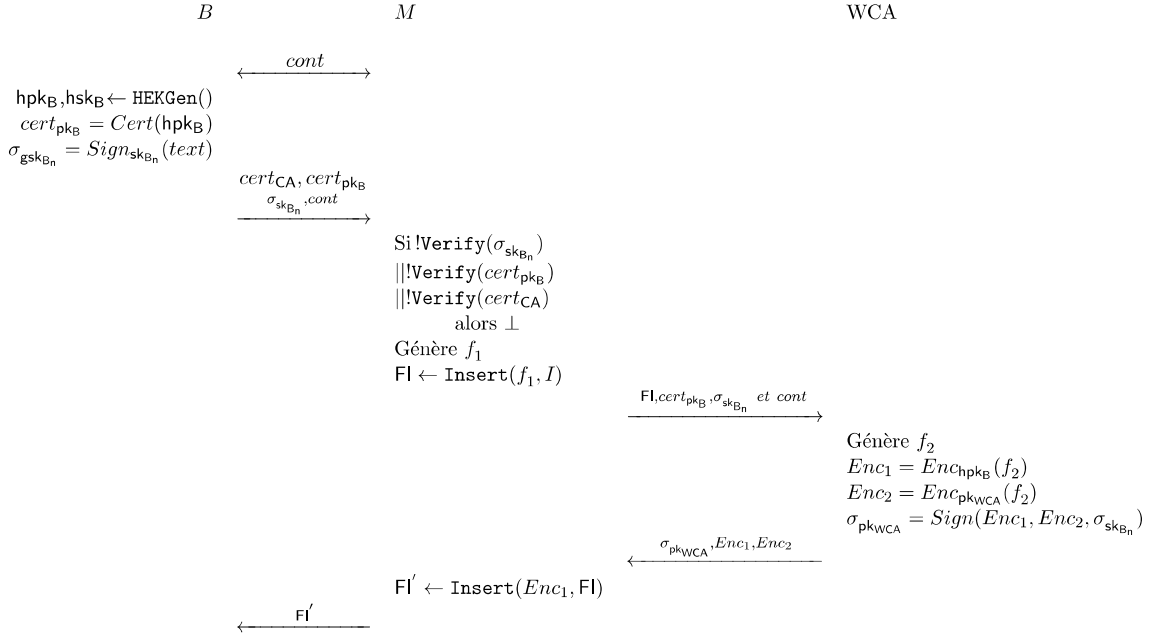

FIGURE 2.11.: Phase d'achat de la solution de Lei *et co-auteurs* [81]

Lors de l'identification (figure 2.12), le vendeur récupère les informations de l'acheteur ayant le f_1 le plus proche de $f_1^{'}$ (extrait de la copie pirate FI_*). Le vendeur envoie ensuite les informations disponibles pour cette vente et pour cet acheteur à J. Plus précisément, les informations fournies au juge sont $cert_{\mathsf{CA}}, cert_{\mathsf{pk_B}}, \sigma_{\mathsf{sk_{B_n}}}$ et $cont$. J est chargé de vérifier les signatures et les certificats. S'ils sont tous valides, alors J demande au WCA de déchiffrer Enc_2 pour obtenir f_2. Ensuite, il vérifie enc_1 en chiffrant f_2 avec $\mathsf{hpk_B}$. Le juge obtient $enc_1^{'}$ et compare enc_1 et $enc_1^{'}$. Si la vérification réussit, alors J cherche f_2 dans FI^*. Si f_2 est trouvé, alors l'acheteur est accusé ; sinon l'acheteur n'est pas exposé. Cependant, en cas de collusion, f_2 ne sera pas trouvée, car ils ne prennent pas en compte les empreintes proches. De ce fait, leur solution n'est pas résistante aux collusions sur f_2 donc la probabilité de retrouver un acheteur est faible. Un inconvénient majeur de cette solution est qu'une tierce partie de confiance est utilisée pour générer f_2. Cela implique qu'en cas de compromission de la tierce partie de confiance l'adversaire aura connaissance de toutes les empreintes f_2.

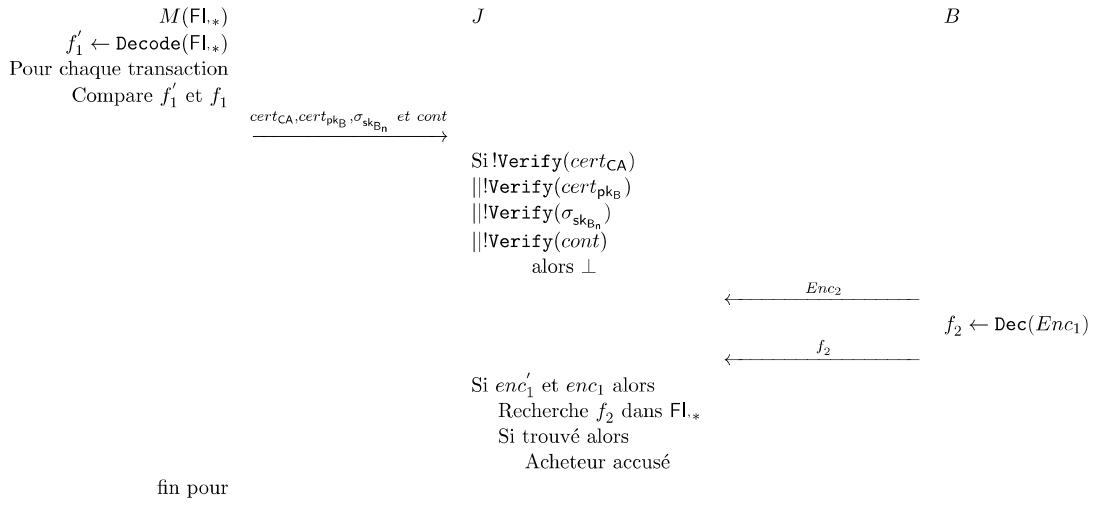

FIGURE 2.12.: Phase d'accusation de la solution de Lei *et co-auteurs* [81]

Abdul *et co-auteurs* [12] ont proposé un protocole anonyme de personnalisation de contenus en 2010. Ils utilisent un protocole 2-parties entre le vendeur et l'acheteur, que cela soit pour la génération et l'intégration de l'empreinte ou pour les poursuites engagées contre l'acheteur. Leur protocole utilise les primitives cryptographiques suivantes :

— Envoi superposé (*superposed sending*) : utilisé pour masquer qui est l'acheteur ayant effectué la requête.
— Chiffrement homomorphe : utilisé dans la primitive de retrait d'information privée (PIR).
— Signature de groupe : utilisé pour masquer l'identité de l'acheteur.
— Retrait d'information privée : utilisé pour que le vendeur ne sache pas quel est le contenu utilisé.

L'acheteur B_n est supposé avoir un couple de clefs publique/privée $(\mathsf{gpk}_{\mathsf{B_n}},\mathsf{gsk}_{\mathsf{B_n}})$ d'un schéma de signature de groupe et deux clefs secrètes $\mathsf{sk}_{\mathsf{B_n}}$ et $\mathsf{sk}_{\mathsf{B_n}}{}'$ d'un schéma de chiffrement symétrique. De plus, tous les envois se font à l'aide d'envois superposés, dont le but est de cacher l'émetteur du message. En effet, lorsqu'un acheteur désire envoyer un message, tous les acheteurs du groupe envoient une chaîne de caractère 0 au vendeur sauf l'acheteur qui envoie la requête. L'envoi superposé de la part du vendeur consiste à envoyer le message à tous les membres du groupe, mais seul le destinataire légitime traitera ce dernier.

Dans un premier temps, l'acheteur B_n envoie une requête R au vendeur M pour acheter un contenu I. Cette requête contient I, ainsi que sa clef symétrique $\mathsf{sk}_{\mathsf{B_n}}$ et sa signature $\sigma^B_{I_{\mathsf{gsk}_{\mathsf{B_n}}}}$. Ensuite, le vendeur vérifie $\sigma^B_{I_{\mathsf{gsk}_{\mathsf{B_n}}}}$ avec gpk. Si la vérification échoue, il sort $\perp$; sinon il continue. Ensuite, le vendeur chiffre avec sa clef privée sk_{M} le couple $(N_T,\sigma^B_{I_{\mathsf{gsk}_{\mathsf{B_n}}}})$ et obtient $Enc_{\mathsf{sk}_{\mathsf{M}}}(N_T,\sigma^B_{I_{\mathsf{gsk}_{\mathsf{B_n}}}})$. Il utilise alors $\mathsf{sk}_{\mathsf{B_n}}$ pour chiffrer ce résultat et il obtient $Enc_{\mathsf{sk}_{\mathsf{B_n}}}(Enc_{\mathsf{sk}_{\mathsf{M}}}(N_T,\sigma^B_{I_{\mathsf{gsk}_{\mathsf{B_n}}}}))$. L'acheteur déchiffre le message avec $\mathsf{sk}_{\mathsf{B_n}}$ et après avoir ajouté son identité Id_{B_n}, il chiffre le nouveau message avec $\mathsf{sk}_{\mathsf{B_n}}{}'$. Le vendeur ajoute maintenant son identité Id_M et insère $(Enc_{\mathsf{sk}_{\mathsf{B_n}}{}'}(Enc_{\mathsf{sk}_{\mathsf{M}}}(N_T,\sigma^B_{I_{\mathsf{gsk}_{\mathsf{B_n}}}})+Id_{B_n})+Id_M)$

Si le vendeur retrouve un document redistribué sans autorisation, il extrait l'empreinte, enlève son identité

$$Enc_{\mathsf{sk}_{\mathsf{B_n}}{}'}(Enc_{\mathsf{sk}_{\mathsf{M}}}(N_T,\sigma^B_{I_{\mathsf{gsk}_{\mathsf{B_n}}}})+Id_{B_n})$$

et envoie le reste à tous les acheteurs à l'aide de l'envoi superposé. Celui ayant commis la redistribution illégale déchiffre et envoie les informations déchiffrées

$$Enc_{\mathsf{sk}_{\mathsf{M}}}(N_T,\sigma^B_{I_{\mathsf{gsk}_{\mathsf{B_n}}}})+Id_{B_n}$$

au vendeur. Celui-ci pourra alors déchiffrer la dernière partie

$$Enc_{\mathsf{sk}_{\mathsf{M}}}(N_T,\sigma^B_{I_{\mathsf{gsk}_{\mathsf{B_n}}}})$$

et obtenir l'identifiant de la transaction ainsi que la signature de groupe de l'acheteur. Il pourra alors contacter le juge pour la procédure d'accusation.

Ce protocole possède des inconvénients dont nous discutons ici. Premièrement, l'envoi de la clef symétrique de l'acheteur n'est pas utile, car le retour serait déjà chiffré par la clef publique du vendeur, donc il n'y a pas besoin de chiffrer un chiffré. De plus, le vendeur envoie toujours à tous les acheteurs ce qui est lourd alors qu'il serait possible d'utiliser, pour empêcher de remonter à la source du message, un protocole de communication anonyme comme TOR. Aussi, lorsque l'acheteur reçoit la demande de déchiffrement de l'empreinte par le vendeur, celui-ci n'est pas obligé de répondre et d'obéir au vendeur. En effet, le vendeur ne sachant pas qui a commis la redistribution, l'acheteur ne sera pas inquiété s'il ne répond pas. Enfin, il n'y a pas de résistance aux collusions. Il sera donc impossible dans ces cas-là de retrouver au moins un acheteur malhonnête. Pour finir, l'envoi superposé suppose que les utilisateurs soient toujours en ligne. Ce qui nous paraît irréaliste dans la pratique.

Malgré ces inconvénients, un avantage, comparé à toutes les solutions précédentes est de cacher quel contenu est récupéré par l'acheteur au vendeur (non-chaînabilité des contenus). Cette propriété est atteinte par l'utilisation de Retrait d'Information Privé (décrit au chapitre 2 section 2.3.3). Le retrait d'information privée est quant à lui effectué à l'aide de chiffrement homomorphe.

Rial, Balasch et Preneel ont également proposé en 2011 un protocole permettant de cacher le contenu acheté au vendeur [112](*non-chaînabilité des contenus*). Contrairement aux articles présentés ci-dessus, ce dernier ne prend pas en charge l'anonymat des acheteurs. En effet, l'acheteur est authentifié auprès du vendeur et seul le contenu récupéré par l'acheteur est masqué d'un point de vue du vendeur. Cette solution est basée sur un POT ou Transfert Inconscient avec Paiement (*Priced Oblivious Transfer*) [15, 114]. L'avantage de cette solution est qu'il est possible de cacher le contenu récupéré par l'acheteur même si les prix des contenus proposés par le vendeur sont différents.

Alfredo Rial *et co-auteurs* [113] ont proposé un protocole de personnalisation de contenus anonyme et des preuves de son efficacité. Contrairement aux précédentes solutions de Pfitzmann *et co-auteurs* celle-ci est composée de cinq entités. Ils reprennent les quatre entités de Pfitzmann et ajoutent une autorité de lever d'anonymat (OA) chargée de révoquer l'anonymat des acheteurs uniquement sur requête du juge. Ils utilisent les concepts cryptographiques de signature de groupe, de chiffrement homomorphe et de preuve à divulgation nulle de connaissance. Leur protocole contient quatre sous protocoles :

1. `Setup` : cette phase consiste en la génération de la clef privée de CA ($\mathsf{sk}_{\mathsf{CA}}$), de la clef de lever d'anonymat $\mathsf{sk}_{\mathsf{OA}}$ et de la clef publique de signature de groupe gpk. L'acheteur B_n et le juge J génèrent une paire de clefs chacun ($\mathsf{pk}_{\mathsf{B_n}}$,$\mathsf{sk}_{\mathsf{B_n}}$), respectivement ($\mathsf{pk}_{\mathsf{J}}$, sk_{J}). Le vendeur M génère quant à lui une clef de tatouage swk.
2. `BReg` : la phase d'enregistrement se déroule entre CA et B_n qui obtient $\mathsf{gsk}_{\mathsf{B_n}}$. L'autorité d'enregistrement obtient une information d'enregistrement qui est stockée dans une table d'enregistrement.
3. `IBuy` : Cette phase est exécutée lorsqu'un acheteur désire recevoir un contenu

I (Figure 2.13). L'acheteur génère une paire de clefs pour le chiffrement homomorphe ($\mathsf{hpk}_{\mathsf{B_n}}$,$\mathsf{hsk}_{\mathsf{B_n}}$) et il chiffre $\mathsf{hsk}_{\mathsf{B_n}}$ avec $\mathsf{pk_J}$ (Enc_1=$Enc_{\mathsf{pk_J}}(\mathsf{hsk}_{\mathsf{B_n}})$). Ensuite, B_n génère $f_{B_n}^{m_2}$ une empreinte f_{B_n} de taille m_2 sur l'alphabet $\chi = \{0,1\}$ ($f_{B_n}^{m_2} \xleftarrow{Random} \{0,1\}^{m_2}$) avant de chiffrer avec $\mathsf{hpk}_{\mathsf{B_n}}$ chaque bit de $f_{B_n}^{m_2}$ ($Enc_2^{m_2}$) et de signer un message msg ($\sigma_{\mathsf{gsk}_{\mathsf{B_n}}}$=$Sign(msg)$) où msg est composé de $\mathsf{hsk}_{\mathsf{B_n}}$,$Enc_1$ et $Enc_2^{m_2}$. B_n envoie msg et $\sigma_{\mathsf{gsk}_{\mathsf{B_n}}}$ à M. L'acheteur et le vendeur utilisent une preuve à divulgation nulle de connaissance pour que l'acheteur prouve qu'il a bien suivi le protocole. Si M est convaincu, alors il génère aléatoirement $f_M^{m_2}$ et chiffre bit par bit $f_{B_n}^{m_2} \oplus f_M^{m_2}$ avec $\mathsf{hpk}_{\mathsf{B_n}}$. De plus, une chaîne unique et aléatoire ϕ de taille m_1 est générée par M. Il chiffre aussi chaque bit de ϕ avec $\mathsf{hpk}_{\mathsf{B_n}}$. L'empreinte insérée dans le contenu est $f = \phi||(f_{B_n}^{m_2} \oplus f_M^{m_2})$. L'algorithme `Insert` est utilisé pour insérer f dans I dans le domaine chiffré. Le contenu final étant chiffré avec $\mathsf{hpk}_{\mathsf{B_n}}$ et une partie de f étant généré parB_n, seul B_n entre en possession de FI et M ne connaît ainsi que (ϕ,$f_M^{m_2}$,msg,$\sigma_{\mathsf{gsk}_{\mathsf{B_n}}}$).

4. `Accuse` : lorsque le vendeur trouve une copie pirate $\mathsf{FI}_{*,\mathsf{I}}$ de I, il exécute l'algorithme `Accuse`. Tout d'abord, le vendeur utilise `Decode` pour extraire l'empreinte $f = (\phi||x)$. Ensuite, pour chaque entrée stockée (ϕ,$f_M^{m_2}$,msg,$\sigma_{\mathsf{gsk}_{\mathsf{B_n}}}$), M calcule $f_{B_n} = f_M \oplus x$ et obtient en sortie ($f_{B_n}^{m_2}$, msg,$\sigma_{\mathsf{gsk}_{\mathsf{B_n}}}$) ; il envoie les informations au juge. Le juge vérifie alors la signature $\sigma_{\mathsf{gsk}_{\mathsf{B_n}}}$ et arrête le protocole s'il obtient $\perp$. Sinon, il déchiffre Enc_1 et obtient $\mathsf{hsk}_{\mathsf{B_n}}$. Puis, pour chaque bit de $Enc_2^{m_2}$ il déchiffre et obtient $f_{B_n}^*$. Si $f_{B_n}^*$=f_{B_n}, alors il obtient un message $M_{open} = (msg, \sigma_{\mathsf{gsk}_{\mathsf{B_n}}})$ et l'envoie à l'Autorité d'ouverture (OA) qui va désanonymiser l'acheteur malveillant B_n et calculer une preuve π que la désanonymisation est exacte à J. J vérifie π et accuse B_n s'il est convaincu. À chaque vérification, si l'algorithme sort $\perp$, le protocole est arrêté et l'acheteur n'est pas accusé.

Dernièrement, certaines solutions proposées [109, 90] s'appuient sur un réseau pair-à-pair dans le but d'améliorer le temps d'exécution de la phase `IRecover`. Pour [109], une tierce partie de confiance génère un code de Tardos, le chiffre et l'envoie au vendeur qui l'insère dans un sous-ensemble du contenu (fichier de base) dans le domaine chiffré et ensuite l'envoie au client. Le reste du contenu (fichiers supplémentaires) n'est pas marqué et est distribué par des nœuds coordinateurs qui gèrent des groupes de nœuds (acheteur). Ces nœuds coordinateurs possèdent tous les fichiers supplémentaires des contenus que vend le vendeur. Lors de la récupération des fichiers supplémentaires, l'acheteur demande à son nœud coordinateur de les lui fournir. Si ce dernier ne le possède pas, il demande à tous les acheteurs de son groupe s'ils le possèdent. Si oui, il fait la liaison entre l'acheteur ayant émis la requête et les autres. Les acheteurs sont authentifiés anonymement entre eux pour l'échange. S'ils ne l'ont pas, alors le nœud coordinateur demande aux autres nœuds coordinateurs du réseau. Dans ce protocole, le vendeur ne fournit qu'une partie du contenu (partie marquée) et le reste est fourni par d'autres nœuds du réseau. La tierce partie de confiance étant la seule à connaître les mots de codes doit impérativement participer à l'accusation. Les acheteurs et le vendeur font confiance à cette tierce partie. Le fait qu'une tierce partie, même de

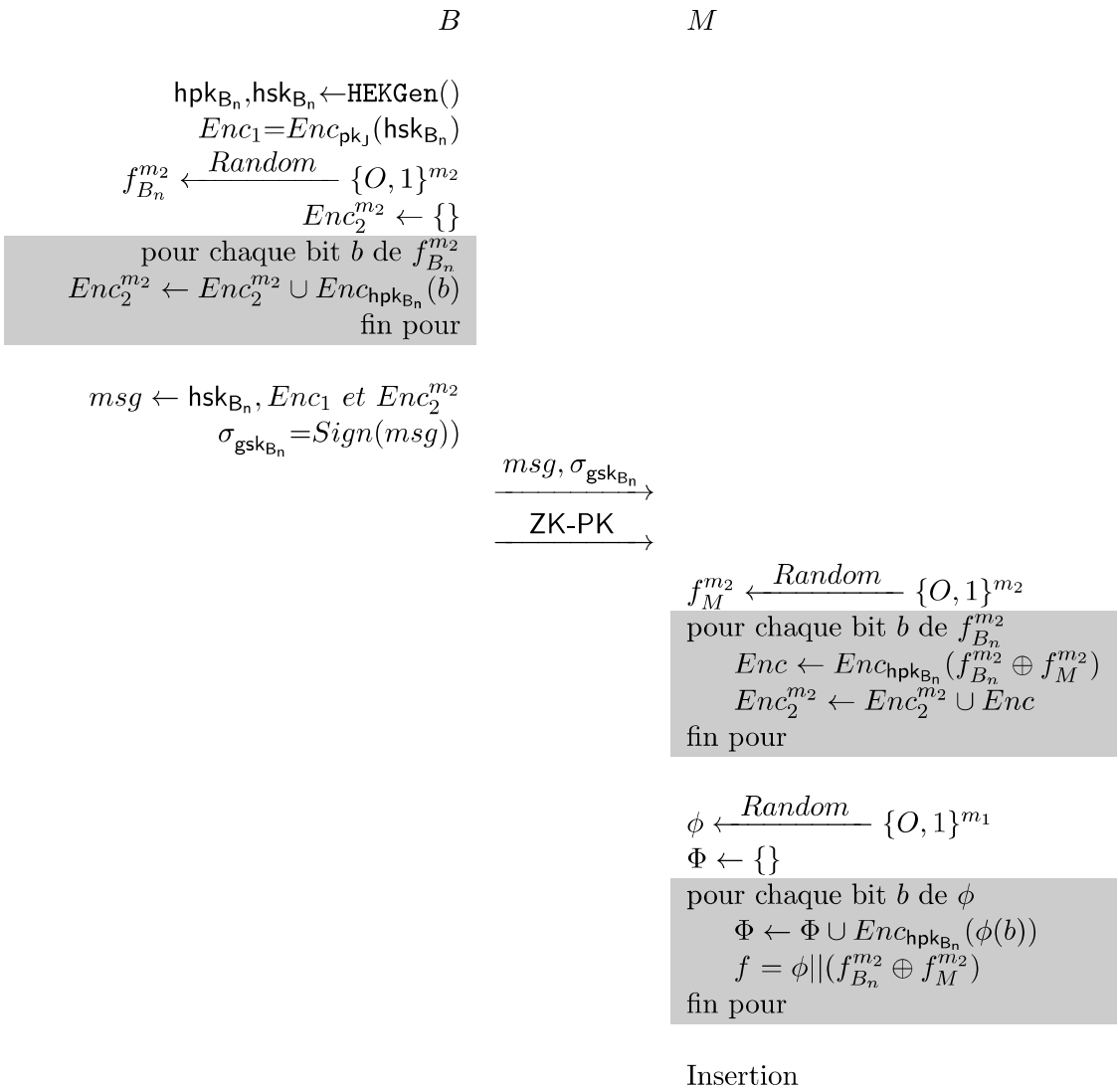

FIGURE 2.13.: Phase d'achat (`IBuy`) des acheteurs de la solution de Rial *et co-auteurs* [113]

confiance, possède tous les mots de code n'est pas recommandé. En effet, si celle-ci est compromise alors l'adversaire connaîtra tous les mots de codes des utilisateurs. De plus, le vendeur ne fournissant que certains blocs marqués à l'acheteur diminue sa capacité de personnalisation. En effet, moins de blocs sont disponibles moins de bits pourront être insérés. Cela implique que l'empreinte sera plus courte et que la probabilité que deux acheteurs aient le même mot de code augmente.

Dans le tableau 2.6 nous présentons les propriétés respectées par les solutions de personnalisation de contenus préservant la vie privée que nous venons de décrire. Comme nous pouvons le voir, tous les protocoles présents dans ce tableau ne respectent pas

	[106]	[103]	[25]	[113]	[112]	[12]	[49]	[81]
Anti-Framing	V	V	V	V	V	V	V	V
Traitor tracing	(Oui)	(Oui)	(Oui)	(Oui)	(Oui)	Non	(Oui)	(Oui)
Analyse de sécurité formelle	X	X	V	V	V	X	X	X
Anonymat révocable	V	V	V	V	X	V	V	V
Non-chaînabilité des acheteurs	X	V	V	V	X	V	X	X
Non-chaînabilité des contenus	X	X	X	X	V	V	X	X
Compatible avec les codes de Tardos	X	X	X	X	X	X	X	X
Implémentation	X	X	X	V	V	X	X	X

Tableau 2.6.: Respect des propriétés de sécurité et protection de la vie privée par les protocoles présents dans la littérature. La couleur orange concernant le traçage de traîtres indique, soit que les codes de ces solutions sont trop grands pour être utilisés en pratique, soit que le code est inefficace contre les collusions. En vert, ce sont les codes utilisables en pratique et permettant de contrer des collusions.

tous les critères de comparaisons et toutes les propriétés de sécurité et de respect de la vie privée définies.

2.4.3. Personnalisation de bases de données assainies

Très peu de travaux ont été effectués sur la combinaison de techniques de tatouage/personnalisation de bases de données avec le respect de la vie privée. Nous présentons succinctement l'état de l'art ci-dessous.

En 2005, Bertino, Beng, Yanjiang et Deng [18] ont proposé un schéma permettant de garantir le respect de la vie privée des individus présents dans la base de données tout en tatouant celle-ci pour la preuve de propriété. Ainsi, la marque présente dans la base de données ne permet pas de tracer des traîtres, mais permet de prouver qui est son propriétaire. Dans cette solution, les auteurs utilisent d'abord du k-anonymat pour assainir la base de données (à l'aide des procédés de généralisation et de suppression) avant de marquer celle-ci. Le marquage est vu comme une permutation des données et est réalisé par l'utilisation de clefs secrètes.

Schrittwieser, Kieseberg, Echizen, Wohlgemuth, Sonehara et Weippl ont proposé un algorithme de k-anonymat pour conduire la personnalisation dans [118]. Tout d'abord, ils recherchent toutes les façons de k-anonymiser la base de données $\mathcal{D}$ en accord avec le k choisi. Pour chaque solution trouvée, la précision des données est mesurée. Cette mesure a pour objectif de s'assurer que les données sont assainies, mais toujours utilisables. L'empreinte d'un utilisateur est constituée des niveaux de généralisation par attribut composant la base de données envoyée à l'acheteur. Dans la pratique, le nombre de solutions possibles, et donc le nombre d'empreintes réalisables, est limité par k, la

mesure de précision et la taille de la base de données (nombre de tuples et d'attributs). Cependant, le k-anonymat étant sensible à certaines attaques (voir section 2.3.2), il ne nous apparaît pas judicieux de s'en servir en pratique.

2.5. Identification et respect de la vie privée : paradoxe ?

Le respect de la vie privée est, comme dit dans la section 2.1, le fait de protéger toute information à caractère personnel permettant l'identification directe ou indirecte d'un (ou plusieurs) individu(s). Par opposition, l'identification est le fait de pouvoir connaître l'identité d'un (ou plusieurs) individu(s). Dans cette thèse, nous parlons des protocoles de personnalisation de contenus permettant de tracer les traîtres tout en protégeant la vie privée des utilisateurs honnêtes. En d'autres termes, nous devons garantir que la vie privée des utilisateurs ne puisse pas être bafouée (c'est-à-dire, qu'aucune information personnelle ne sera connue du vendeur), mais nous devons être capables d'identifier les utilisateurs malhonnêtes. D'un point de vue général, considérant que la protection de la vie privée est un droit fondamental, il nous paraît nécessaire de proposer des solutions protégeant celle-ci tout en garantissant que le nouveau protocole fonctionne aussi efficacement que ceux qui ne la préservent pas. Cependant, il est aussi nécessaire de pouvoir identifier et poursuivre les utilisateurs malveillants, car le droit à la vie privée n'est pas un droit à la malveillance. De ce fait, lorsque nous parlons d'anonymat il s'agit d'anonymat révocable. De plus, bien que vouloir garantir simultanément l'identification et le respect de la vie privée puisse paraître paradoxal, les protocoles de personnalisation de contenus préservant la vie privée des utilisateurs font sens. En effet, l'utilisation de tels protocoles permet, d'un côté, d'assurer aux fournisseurs de contenus numériques que les utilisateurs malveillants pourront être identifiés et d'un autre côté de garantir aux utilisateurs honnêtes leur droit fondamental au respect de leur vie privée. Ces protocoles peuvent s'appliquer à la distribution de films, chansons, ... en version numérique, tout en conservant la valeur commerciale et l'utilisation légale de ces contenus. Considérons l'exemple d'un système de distribution de films numériques pouvant utiliser un protocole de personnalisation anonyme. Il existe, par exemple, dans certains boîtiers de DVD, un code permettant de récupérer le film, contenu dans la boite, pour le conserver sur un disque dur. La récupération ne se fait qu'une fois que l'utilisateur a créé un compte chez le fournisseur du contenu. Il pourrait être intéressant de permettre à l'utilisateur de récupérer son film de manière anonyme tout en garantissant le respect du droit d'auteur, y compris en cas de redistribution illégale du contenu.

Deuxième partie .

Contributions

3. Protocole de personnalisation de contenus asymétrique préservant la vie privée des acheteurs

Les protocoles de personnalisation de contenu ont été originellement inventés pour dissuader les utilisateurs malveillants de distribuer un contenu numérique illégalement, tels qu'une image, un film ou encore une chanson. Pour faire cela, à chaque fois qu'un contenu est distribué, une marque unique est insérée dans celui-ci. Si celle-ci est créée à l'aide d'un code anti-collusion, le protocole pourra tracer une collusion d'utilisateurs qui ont forgé une fausse copie (c'est-à-dire, une copie pirate) d'un contenu à partir de leurs copies légitimes. Charpentier, Fontaine, Furon et Cox [33] ont été les premiers à proposer un protocole de personnalisation de contenu asymétrique reposant sur les codes de Tardos. Une autre solution a été proposée par Kiayias, Leonardos, Lipmaa, Pavlyk et Tang [73], en 2015. Les codes de Tardos sont les codes anti-collusion les plus efficaces connus à ce jour. C'est-à-dire qu'à taille de collusion maximale tolérée par le système, les codes de Tardos sont les plus courts, ce qui leur permet d'être utilisés en pratique. Kiayias *et co-auteurs* se sont concentrés sur les propriétés de sécurité nécessaires pour un tel protocole, mais ce dernier ne préserve pas la vie privée des différents utilisateurs. Pour résoudre ce problème, nous introduisons dans ce chapitre PIMENTO, qui est le premier protocole de personnalisation de contenu asymétrique reposant sur les codes de Tardos et préservant la vie privée des utilisateurs. Ce protocole est optimal en ce qui concerne le traçage de traîtres, c'est-à-dire, que nous retrouvons un traître avec une forte probabilité pour chaque contenu diffusé illégalement. Nous définissons dans ce chapitre, tout d'abord informellement dans la section 3.1, les propriétés de sécurité et de protection de la vie privée que nous voulons atteindre puis formellement dans la section 3.3. Dans la section 3.2, nous présentons notre protocole (PIMENTO) ainsi que son amélioration (PIMENTO+) au niveau de la protection de la vie privée. Ensuite, nous prouvons, dans la section 3.4 que PIMENTO assure les propriétés définies avant de montrer des résultats en terme d'efficacité de calcul dans la section 3.5. Cette contribution a fait l'objet d'une publication à LatinCrypt 2014 [53] et un article journal est en cours de soumission.

3.1. Modèle de sécurité

Dans cette section, nous commençons par décrire les modèles du système et de l'adversaire. Ensuite, nous expliquons les hypothèses d'insertion et de marquage dont nous avons besoin avant de définir les exigences de sécurité et de vie privée pour les proto-

coles de marquage de document asymétrique préservant la vie privée. Cette section est écrite de manière informelle. La version formelle du modèle de sécurité, de l'adversaire et des propriétés est présentée dans la section 3.3.

3.1.1. Modèles du système et de l'adversaire

Nous considérons un système composé des parties suivantes :

— Un ensemble d'*acheteurs*, enregistrés auprès d'une autorité de certification, pouvant recevoir un contenu personnalisé auprès du vendeur après une transaction.
— Un seul *vendeur* qui possède un ensemble de contenus numériques et pouvant personnaliser ceux-ci et autoriser, après une transaction réussie, un acheteur à récupérer une copie personnalisée de ce contenu.
— Une *autorité de certification* (CA) enregistrant les acheteurs dans le système. Nous supposons que les accréditations du vendeur sont certifiées (pas obligatoirement auprès du même CA).
— Une *autorité d'ouverture* (OA) qui lève l'anonymat des acheteurs suspects sur présentation d'une preuve par le vendeur. L'identité révélée des suspects n'est pas communiquée au vendeur.

Nous supposons que toutes les communications entre le vendeur et les acheteurs ont lieu à l'aide d'un canal de communication anonyme (tel que le réseau TOR). De plus, si l'acheteur doit payer pour un contenu, nous exigeons que la transaction soit anonyme. Ces hypothèses sont nécessaires, mais pas suffisantes. En particulier, comme montré dans [75], un canal de communication anonyme ne fait que préserver la vie privée, mais ne la crée pas. En effet, le contenu du message doit aussi assurer la vie privée de l'émetteur et du récepteur. Dans notre contexte, nous sommes seulement concernés par la protection de la vie privée des acheteurs. Nous avons aussi besoin que l'acheteur reçoive sa clef privée de CA à l'aide d'un canal sécurisé (c'est-à-dire, authentifié et confidentiel).

Hypothèses d'insertion et de personnalisation de contenus. Nous rappelons qu'un protocole est exécuté entre l'acheteur et le vendeur chaque fois que le premier veut récupérer un contenu spécifique. À la fin de ce protocole, l'acheteur récupère le contenu tel que chaque bloc est marqué avec exactement un bit. De plus, chaque version récupérée par un acheteur est personnalisée avec une empreinte unique. Nous supposons que l'empreinte, insérée par un algorithme de tatouage, est imperceptible par un humain et robuste à certaines attaques comme détaillées ci-dessous.

1. La fonction de tatouage, notée W, n'autorise pas les adversaires à retrouver ne serait-ce qu'un bit de l'empreinte.

2. La technique de tatouage est robuste par rapport aux attaques sur le signal, telles que compression, impression, scan et redimensionnement.

3. Dans une *attaque par collusion* (présentée dans la section 1.4 du chapitre 1), une collusion d'acheteurs malveillants utilise des parties de leurs copies pour forger un contenu. Plus précisément, cela peut conduire à la combinaison des bits de leurs identifiants de manière arbitraire, voir introduire des effacements ou des erreurs,

sous la seule restriction de la *Marking assumption*. La *marking assumption* établit que les membres de la collusion sont restreints par le fait que s'ils ont le même bloc marqué à la même position dans toutes leurs copies, ils ne peuvent pas forger une copie dans laquelle le bit de l'empreinte à cette position spécifique est différent de celui de leurs copies. Si la collusion ne concerne qu'un seul acheteur, cette hypothèse exclut que cet acheteur puisse produire une empreinte forgée différente de l'empreinte présente dans son contenu.

Insertion et tatouage de contenu. Nous utilisons un schéma de tatouage de contenu robuste tel que [134], ce qui permet de contrer les attaques possibles sur le signal. Ceci permet de contrer les attaques sur le signal.

3.1.2. Propriétés de PIMENTO et de PIMENTO+

Dans cette section, nous présentons les propriétés de sécurité et de protection de la vie privée que notre protocole doit garantir.

Conformité. Si tous les participants du protocole sont honnêtes et réalisent chaque étape de la manière attendue alors la sortie du protocole est conforme aux spécifications.

Propriétés de sécurité.

Anti-framing. Cette propriété garantie à chaque acheteur qu'aucun vendeur malveillant ne pourra l'accuser à tort s'il n'a commis aucun acte répréhensible. De plus, nous assurons la propriété de disculpation, c'est-à-dire qu'un acheteur ne pourra être accusé à tort à cause d'une collusion d'acheteurs malveillants.

Traçage de Traîtres. Le traçage de traîtres garantit au vendeur qu'il pourra accuser au moins un acheteur malveillant à chaque redistribution illégale d'un contenu, et ce, avec une forte probabilité de retrouver un traître.

Propriétés de protection de la vie privée.

Anonymat révocable. Cette propriété garantie qu'un acheteur restera anonyme aussi longtemps qu'il sera honnête (c'est-à-dire, temps qu'il ne redistribuera pas illégalement un contenu) et son anonymat sera levé dans le cas contraire.

Non-chaînabilité des acheteurs. Garantie aux acheteurs qu'un vendeur ne peut pas distinguer si deux transactions sont liées ou non au même acheteur.

Non-chaînabilité des contenus. Cette propriété est ajoutée à PIMENTO pour obtenir PIMENTO+. Le vendeur n'apprend pas quels sont les contenus vendus au moment des transactions réalisées avec les acheteurs à condition que le vendeur propose plus d'un contenu à la vente.

3.2. Protocole de personnalisation de contenus préservant la vie privée et reposant sur les codes de Tardos

Dans le but d'assurer les fortes propriétés de sécurité et de vie privée décrites dans la section 3.3, nous introduisons deux nouveaux protocoles de personnalisation de contenu préservant la vie privée et reposant sur les codes de Tardos : PIMENTO et PIMENTO+. Nos solutions étendent le protocole de personnalisation de contenu asymétrique utilisant les codes de Tardos de Charpentier, Fontaine, Furon et Cox [33]. Nos protocoles utilisent des primitives cryptographiques telles que : les signatures de groupe, le transfert équivoque et des preuves de connaissance à divulgation nulle de connaissance non interactive pour obtenir les différentes propriétés que nous avons identifiées pour PIMENTO (section 3.2.1) et pour PIMENTO+ (section 3.2.2). Dans PIMENTO+, nous supposons que les prix des contenus sont égaux. La différence entre PIMENTO et PIMENTO+ réside dans le fait que PIMENTO+ atteint la propriété de non-chaînabilité des contenus. Dans le cas, où les prix seraient différents, il est possible d'utiliser le problème des sous-ensembles (Subset Sum Problem [88] en anglais) et les acheteurs pourraient payer par exemple à la fin de chaque mois. Ce problème est basé sur le fait qu'il serait impossible au vendeur de trouver un sous-ensemble de l'ensemble des tarifs dont la somme des valeurs est égale au prix payé par l'acheteur. Le problème qui se pose avec cette solution réside dans le fait de prouver au vendeur qu'un utilisateur anonyme a payé. Une manière de corriger se problème serait d'utiliser, à la place du problème des sous-ensembles, un transfert équivoque avec paiement (ou Priced Oblivious Transfer [15, 114] en anglais), c'est-à-dire que pour chaque message reçu par l'acheteur, celui-ci paye le message. Ceci est fait de telle sorte que le vendeur sait que l'utilisateur a payé le message au tarif indiqué, mais ne sait pas combien il a payé.

3.2.1. Personnalisation de contenu asymétrique et anonyme

PIMENTO consiste en six phases : Initialisation (`Setup`), Enregistrement (`BReg`), Préparation des contenus (`IPrep`), Achat (`IBuy`), Récupération d'un contenu (`IRecover`), et Accusation et Ouverture (`Accuse` et `Open`). Ces phases sont décrites dans les figures 3.1 à 3.9. Dans la suite de cette section, nous décrivons chaque phase d'abord de manière informelle puis de manière formelle.

`Setup`. La phase d'*initialisation* fournit les clefs nécessaires au CA et à l'OA. Plus précisément, durant la procédure `Setup`, l'algorithme `GSKGen` est exécuté et fournit en sortie la clef maître $\mathsf{sk}_{\mathsf{CA}}$ d'enregistrement des acheteurs et la clef de levée d'anonymat (ouverture des signatures de groupe) $\mathsf{sk}_{\mathsf{OA}}$. Ces clefs sont respectivement données à CA et à OA. CA génère aussi une clef publique de signature de groupe $\mathsf{pk}_{\mathbb{G}}$ pour la vérification des signatures.

`BReg`. Les acheteurs s'enregistrent auprès du CA et reçoivent une clef secrète de signature de groupe[24]. Formellement, nous supposons que le CA garde une trace des

24. Les schémas de signature de groupe fournissent l'anonymat des acheteurs pour chaque transaction sans tenir compte du fait que notre protocole possède ou non un moyen de paiement électronique

acheteurs malveillants et qu'il communique avec les différents acheteurs à travers un canal de communication anonyme. Durant la procédure `BReg`, quand l'identité de B_i est fournie en entrée, le CA vérifie si cet acheteur est sur liste noire BL (si cela est le cas alors le CA produit en sortie le symbole spécial $\perp$). Sinon, le CA transmet une clef secrète de signature de groupe $\mathsf{sk}_{\mathsf{B_i}}$ à l'acheteur. La description de la phase d'enregistrement est décrite dans la figure 3.1.

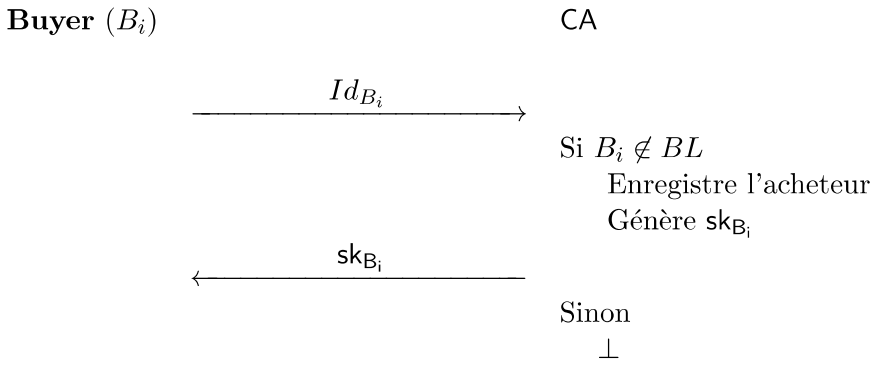

FIGURE 3.1.: Phase d'enregistrement des acheteurs de PIMENTO

`IPrep`. Indépendamment de l'enregistrement des acheteurs, le vendeur exécute la phase de préparation des contenus. Chaque contenu est supposé avoir une longueur uniforme de m blocs (les contenus plus courts sont complétés jusqu'à atteindre la longueur définie). Durant la phase de préparation des contenus, le vendeur génère les paramètres des codes de Tardos (à savoir la probabilité de fausse alarme par utilisateur δ, qui donne directement la limite du nombre maximum c de membres de la collusion pouvant être tolérés). La valeur de δ doit être plus petite que la valeur de fausse alarme certifiée par le CA. Par la suite pour chaque contenu, le vendeur génère un WORM, dans lequel chaque entrée (i, j) correspond au chiffrement du $i^{ème}$ bloc d'une copie marqué du contenu. Plus précisément, le bloc est marqué avec un symbole 0 ou 1, dépendant de la probabilité p_i. Ces probabilités sont générées identiquement et indépendamment de manière aléatoire. Ensuite, le vendeur prouve que le processus de préparation du contenu a été correctement fait en calculant une preuve NIZK-PK. Pour finir, le WORM et la preuve NIZK-PK sont publiés comme décrit dans la figure 3.2. Cette phase est construite sur la base de celle présente dans [33]. La différence majeure est que nous utilisons des blocs tatoués au lieu de symboles de marquage (voir section 3.6.2).

Formellement, durant la phase de préparation des contenus `IPrep` (Fig. 3.3 et 3.5), le vendeur commence par générer les paramètres des codes de Tardos : c, Z et δ. Ces paramètres dépendent de la taille maximum du contenu m. Ensuite, le vendeur génère le vecteur de probabilité $\mathbf{p} := (p_1, \ldots, p_m)$. Puis, pour chaque contenu I_t il procède ainsi :

anonyme. De plus, ils autorisent l'OA à lever l'anonymat d'une signature durant la phase d'accusation.

1. Génération des bits $f_{I_t}^{i,j}$ pour $i \in \{1,\ldots,m\}$ et $j \in \{1,\ldots,N\}$ où j est une ligne de la matrice et i une colonne, qui sont ensuite stockés dans une matrice $\mathcal{F}_{I_t}$ de taille $N \times m$,
2. Génération de $N \times m$ clefs $\mathsf{k}_{I_t}^{i,j}$ d'un schéma de chiffrement symétrique IND-CPA-secure avant de les stocker dans une matrice κ_{I_t} pour chaque contenu I_t,
3. Création du WORM $\tau_{\mathsf{I_t}}$ dont les entrées $\mathsf{fb}_{I_t}^{i,j} = \mathtt{Enc}(\mathsf{k}_{I_t}^{i,j}, \mathsf{W}(I_t^i, f_{I_t}^{i,j}))$ sont chiffrées, avec une clef générée précédemment, d'un bloc marqué, avec un bit spécifique, d'un contenu, et
4. Génération d'un NIZK-PK [25] prouvant que les paramètres des codes de Tardos sont générés correctement (c'est-à-dire, sans triche de la part du vendeur), que $\delta < \delta_{\mathsf{max}}$ et que le WORM est correctement fait. Cette preuve est notée $\pi_{\mathsf{I_t}}$ et générée avec l'algorithme `ProofGen`. Cette phase est détaillée dans les figures 3.4 et 3.5.

Pour chaque contenu, le WORM et la preuve sont publiés. Nous montrons le marquage, la préparation, la transaction et la récupération des contenus dans les Figures 3.2 et 3.3.

De plus, générer le WORM de cette manière permet d'avoir une possibilité de 2^m mots de code ce qui implique que notre protocole supporte au maximum 2^m ventes par contenu. Cependant, les mots de code étant tirés aléatoirement, il existe une probabilité importante de collision, c'est-à-dire que deux acheteurs différents tirent le même mot de code. En effet, il n'y a que deux variantes d'un $i^{ème}$ bloc (un marqué avec un 0 et un marqué avec un 1) par colonne, la probabilité p_i (respectivement $1-p_i$) est la probabilité que l'acheteur tire un bloc marqué avec un 1 (respectivement un 0) pour le $i^{ème}$ bloc du contenu qu'il récupère. Par conséquent la probabilité que deux acheteurs B_1 et B_2 tirent tous les deux le même bloc marqué avec un 1 est p_i^2. De manière similaire, la probabilité que deux acheteurs B_1 et B_2 tirent tout deux un bloc marqué avec un 0 est $(1-p_i)^2$. Nous déduisons que la probabilité que B_1 et B_2 tirent le même symbole (0 ou 1) est égale à $(p_i \times p_i) + (1-p_i) \times (1-p_i)$. Finalement, comme chaque bloc est tiré de manière aléatoire et indépendamment des autres, la probabilité que B_1 et B_2 aient le même symbole en deux positions indépendantes i et j est $(p_i^2+(1-p_i)^2) \times (p_j^2+(1-p_j)^2)$. Par conséquent, la probabilité que deux acheteurs tirent exactement le même mot de code est :

$$\prod_{i=0}^{m}(p_i^2 + (1-p_i)^2) \tag{3.1}$$

De plus, du fait du paradoxe des anniversaires, le nombre maximum de ventes possible pour un contenu est approximativement $\mathcal{O}(2^{\frac{m}{2}})$. Si nous dépassons cette limite,

25. Cette NIZK est une preuve de conformité dans laquelle le témoin se compose du nombre maximum c de membres de la collusion pouvant être détecté, de la probabilité δ qu'un innocent soit faussement accusé, du seuil d'accusation Z et des probabilités p_i pour $i \in \{1,\ldots,m\}$. La déclaration prouvée par la NIZK est constituée des conditions suivantes : $m = 2\pi c^2[\ln \frac{1}{\delta}]$ et que $p_i = (\sin r_i)^2$ pour des r_i aléatoires et uniformément tirés dans un intervalle spécifique (voir [33]).

nous avons une probabilité non négligeable d'avoir une collision, ce qui ferait que deux acheteurs partageraient le même mot de code.

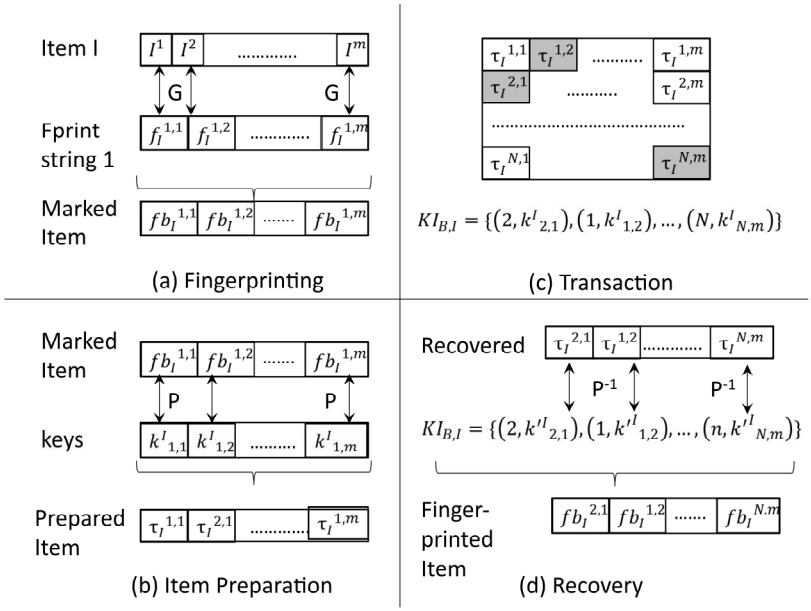

FIGURE 3.2.: Illustration des différentes phases de PIMENTO incluant (a) le marquage du contenu, (b) la préparation des contenus, (c) la transaction entre l'acheteur et le vendeur et (d) la récupération du contenu.

`IBuy`. Le processus d'achat est l'une des phases les plus fondamentales de notre protocole et est composé de deux parties. La première partie consiste à la génération des informations de transaction alors que dans la deuxième, l'acheteur et le vendeur échangent alternativement les rôles d'*envoyeur* et *receveur* au moment de l'exécution de tours d'OT séquentiels (au total m tours). Plus précisément, l'acheteur commence par vérifier la preuve que le WORM est bien formé. Si ce n'est pas le cas, alors l'acheteur arrête le protocole. Sinon, l'acheteur récupère une séquence de m tuples de clefs et les positions des blocs correspondants dans le WORM pour un contenu alors que le vendeur apprend la moitié (en espérance) des indices récupérés par l'acheteur. Ces indices constituent le *halfword* stocké par le vendeur. Pour prévenir les attaques où l'acheteur insère une fausse clef dans l'OT, le vendeur associe, à chaque transaction, un nonce généré aléatoirement à la valeur de la clef. Ce nonce et la clef sont récupérés avec le *halfword*.

Dans notre protocole, ces deux objectifs sont atteints par l'*entrelacement* de tours des OT_1^N et OT_1^2. Plus explicitement, l'acheteur récupère une paire clef/nonce avec un OT_1^N. La clef provient de la colonne correspondante du WORM tandis que le nonce est généré de manière aléatoire à chaque transaction (ici le vendeur joue le rôle de l'envoyeur

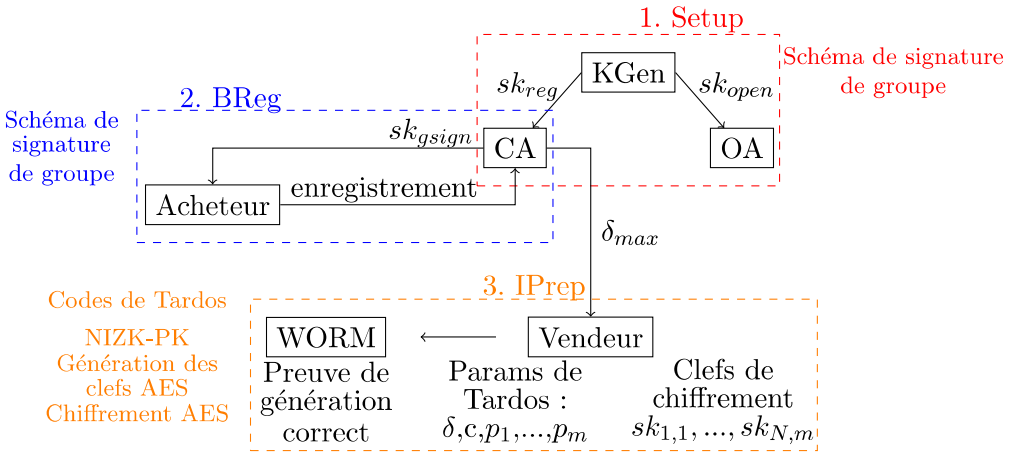

FIGURE 3.3.: Ce schéma montre les différentes interactions entre les entités du protocole pour les phases d'initialisation (en rouge), d'enregistrement (en bleu) et de préparation des contenus (en orange).

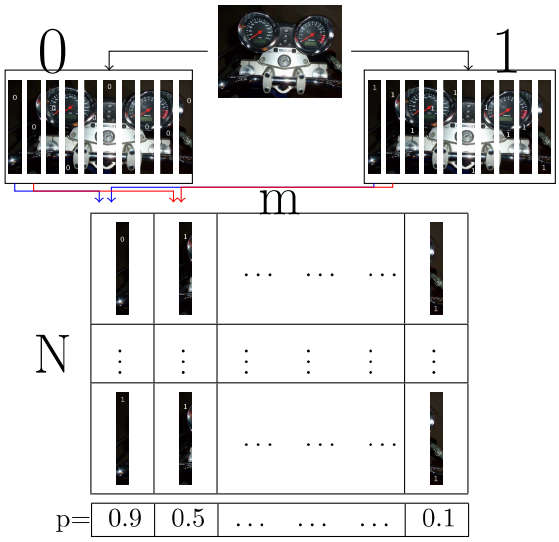

FIGURE 3.4.: Illustration de la phase de préparation des contenus basée sur [33].

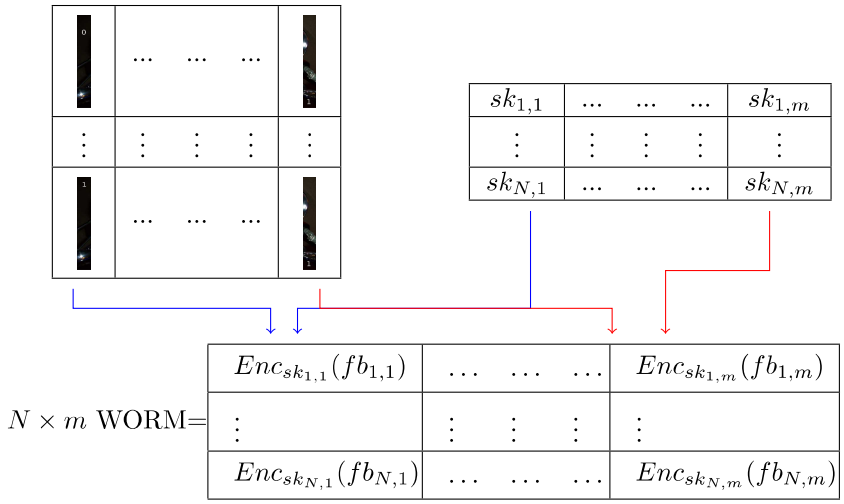

FIGURE 3.5.: Schéma détaillant la construction du WORM.

et l'acheteur celui de receveur). À la fin de ce tour d'OT, un OT_1^2 est exécuté, dans lequel le vendeur récupère l'indice de la clef, la clef et un nonce aléatoire de l'acheteur ainsi qu'une signature de groupe sur la clef et le numéro de la transaction ou un tuple de valeur aléatoire ayant le même format. Ces entrées sont générées aléatoirement par l'acheteur avant de les mettre en entrée de l'OT. De plus, le vendeur n'apprend rien à part le fait que ce qu'il reçoit est bien formé (dans le cas des valeurs clef/nonce) ou non. Ce processus continu en entrelaçant des tours de OT_1^N et de OT_1^2 jusqu'à ce que l'acheteur ait récupéré m tuples d'indice/clef et le vendeur le *halfword* (de taille $\frac{m}{2}$).

Formellement, durant la procédure `IBuy`, le vendeur et les acheteurs communiquent à travers un canal de communication anonyme et authentifié du côté du vendeur[26]. Lorsqu'un acheteur B_i veut acheter un contenu I_t auprès du vendeur, B_i vérifie la preuve $\pi_{\mathsf{I_t}}$ avec l'algorithme `ProofVerif`. Si le WORM n'est pas bien formé, l'acheteur arrête le protocole en produisant en sortie $\perp$. Sinon, le vendeur envoie l'étiquette temporelle TS courante et un numéro de transaction aléatoire t. L'acheteur signe le message msg constitué de ces deux paramètres et du nom du contenu voulu en utilisant sa clef privée de signature de groupe $\mathsf{sk}_{\mathsf{B_i}}$. Ensuite, il envoie le nom du contenu et sa signature $\sigma_{I_t}^{B_i}$ au vendeur qui les vérifie. Si cette vérification échoue, le vendeur arrête le protocole en produisant en sortie un symbole d'erreur $\perp$.

Ensuite, le vendeur génère $N * m$ nombres aléatoires (nonces) $\{R^{i,j}\}$, pour $i \in \{1, \ldots, N\}$ et $j \in \{1, \ldots, m\}$. L'acheteur commence par exécuter un protocole OT_1^N, et récupère une clef $\mathsf{k}_{I_t}^{i,1}$ et la valeur aléatoire correspondante $R^{i,1}$. Ensuite, le vendeur exécute un transfert équivoque OT_1^2 avec l'acheteur et récupère (1) un tuple

26. Dans le but d'atteindre les limites exactes du théorème de la Section 3.4, nous supposons que les acheteurs ont seulement un accès en boîte noire du processus d'achat. Pour plus de détails, voir la remarque 3.1 un peu plus loin.

$(\mathsf{k}_{I_t}^{i,1}, R^{i,1}, \sigma_i)$, tel que σ_i est une signature de groupe sur le tuple $(\mathsf{k}_{I_t}^{i,1}, t)$, où t est le numéro de transaction signé par l'acheteur ou (2) une valeur aléatoire r respectant le format requis. Le vendeur vérifie la signature avant de s'assurer que le tuple $(\mathsf{k}_{I_t}^{i,1}, R^{i,1})$ concorde bien avec son entrée pour le transfert équivoque précédent. Si les deux vérifications réussissent, l'indice i est ajouté au *halfword* Hw, sinon le *halfword* n'est pas changé. Ce processus est répété jusqu'à ce que m clefs soient récupérées par l'acheteur. Le vendeur vérifie à intervalles réguliers qu'il a collecté la moitié des indices des blocs marqués. Si cette condition n'est pas satisfaite, il arrête la transaction. La fréquence de ces vérifications dépend de m. Vu que m est supposément grand, il est raisonnable de s'attendre à ce que le nombre de bits récupérés dans le *halfword* est très proche de la moitié du nombre total de clefs acquises par l'acheteur. À la fin de la transaction, l'acheteur a récupéré un ensemble $\mathbf{KI}_{B,I_t}$ de taille m où les éléments sont des tuples de la forme $(i, \mathsf{k}_{I_t}^{i,1})_{j=1}^{m}$ alors que le vendeur a retrouvé un ensemble Hw d'indices.

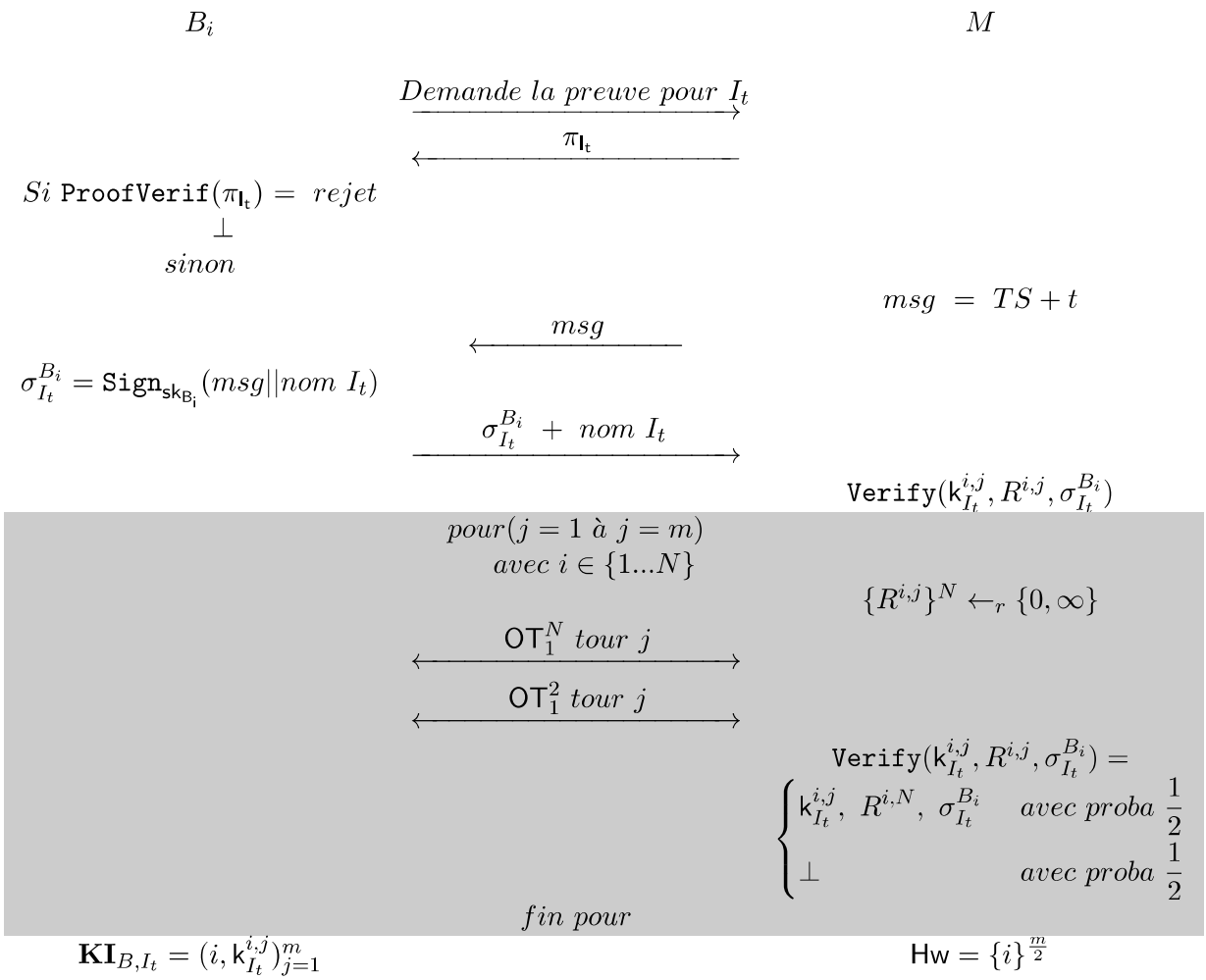

FIGURE 3.6.: Détails de la phase d'achat `IBuy` de PIMENTO.

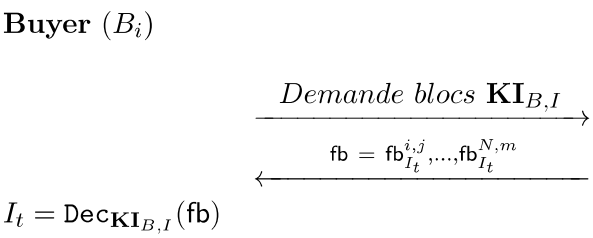

FIGURE 3.7.: Phase de récupération d'un contenu dans PIMENTO.

Durant cette phase, le vendeur ne peut pas choisir le *halfword*, ce qui est essentiel à la propriété d'*Anti-framing*. En effet, si le vendeur est capable de choisir le *halfword* il pourrait apprendre 50% des premiers bits du mot de code d'un acheteur et les 50% derniers bits du mot de code d'un second acheteur. La conséquence serait que le vendeur pourrait utiliser ces deux *halfwords* pour forger une fausse copie qui accuserait ces deux acheteurs.

Aussi, un acheteur ne peut pas utiliser les clefs $\mathsf{k}_{I_t}^{i,1}$ d'un acheteur B_1 à cause des nonces $\{R^{i,j}\}$. En effet, tous les $\{R^{i,j}\}$ ne sont valides que pour une transaction t spécifique. Donc, si un acheteur réutilise une clef, le vendeur verra que le $\{R^{i,j}\}$ ne correspond pas à la clef donnée dans le OT_1^2. Quand cette situation arrive, le vendeur avorte le protocole et sort $\perp$. Les échanges entre le vendeur et l'acheteur durant la phase d'achat sont décrits dans la figure 3.6.

`IRecover`. L'acheteur utilise les indices et les clefs qu'il a obtenus dans le but de récupérer la copie du contenu personnalisée en utilisant le WORM publié pour ce contenu. Spécifiquement, les clefs sont utilisées pour déchiffrer m cellules du WORM avec exactement une entrée par colonne. Formellement, avec l'ensemble des clefs et des indices $\mathbf{KI}_{B,I}$, l'acheteur peut récupérer le contenu personnalisé $\mathsf{FI}_{\mathsf{B,I}}$ avant de le déchiffrer avec les clefs obtenues durant la phase d'achat et de reconstituer le contenu. Cette phase est illustrée dans la figure 3.7.

`Accuse` **et** `Open`. Si le vendeur suspecte qu'une copie numérique d'un contenu spécifique est une copie forgée (résultant d'une collusion de plusieurs acheteurs qui ont mélangé leurs contenus) alors, il calcule un score partiel d'accusation pour chaque transaction du contenu à l'aide du *halfword* qu'il stocke. Si un score partiel dépasse le seuil d'accusation Z_{half1}, cela indique que le signataire de la transaction peut avoir fait une collusion pour forger le contenu. Dans ce cas, un score complet doit être calculé pour tous les acheteurs suspects, afin de décider qui est réellement impliqué dans la collusion. La section 3.6 fournit des détails supplémentaires concernant le calcul des scores d'accusation. Les signatures de transaction, ainsi que les signatures récupérées avec le *halfword* Hw, pour chaque score, avec une preuve NIZK que le score est bien calculé, sont transmises à l'OA. Si la vérification de la NIZK-PK réussie, l'OA ouvre les signatures de groupe reçues du vendeur et retrouve l'identité des acheteurs concernés. Dans PIMENTO l'utilisation de NIZK-PK remplace le premier rôle du juge dans le travail de Charpentier *et co-auteurs* [33], mentionné dans la section 1.6.2. Cela assure que

le vendeur ne triche pas durant le calcul des scores ou lève arbitrairement l'anonymat des acheteurs. Il faut noter que le second rôle du juge, qui consiste à calculer le score sur les empreintes complètes, reste le même et est accompli par l'OA dans notre cas (mais cela peut aussi être délégué à une autre tierce partie de confiance).

Formellement, considérons un contenu forgé $\mathsf{FI}^*_{\mathsf{B}^*,\mathsf{I_t}}$ à partir d'un contenu I_t, durant la phase `Accuse` (figure 3.8), le vendeur calcule les scores partiels d'accusation des codes de Tardos S_t pour chaque transaction t concernant le contenu I_t à l'aide des *halfwords* gardés en mémoire. Plus précisément, le vendeur compare chaque halfword avec le seuil Z_{half1}. Pour chaque score calculé ainsi qui est au-dessous du seuil, le vendeur ajoute à la liste $\mathcal{LO}$ la signature $\sigma^{B_*}_{I_t}$ reçue durant la transaction. Le vendeur prouve la conformité du score d'accusation en utilisant une NIZK-PK. Le témoin pour cette NIZK-PK consiste en la transcription de la transaction pour les parties coupables (incluant les signatures de groupe pour les tours de OT^2_1), leurs scores (calculés comme dans [128]) et le seuil. L'énonce prouvé par la NIZK-PK est que les scores sont calculés correctement, qu'ils sont plus grands que le seuil et que les messages signés envoyés avec la preuve sont bien ceux associés à la transaction. Finalement, le vendeur transmet les indices du contenu forgé I_t, la liste des signatures $\mathcal{LO}$ et la preuve π_{M} à l'OA. Cette phase est détaillée dans la figure 3.8.

Ensuite, pendant la phase `Open` (figure 3.9), si la vérification de la preuve réussit, alors l'OA utilise la clef maître d'ouverture $\mathsf{sk_{OA}}$ pour lever l'anonymat des signatures contenues dans la liste $\mathcal{LO}$. Si le message signé ouvert pour un contenu n'est pas le même que pour le contenu transmis par le vendeur, et pour lequel le calcul a été fait, alors l'OA ne révélera pas l'identité de l'acheteur correspondant. Dépendant du déploiement pratique de notre protocole, le vendeur peut aussi être mis sur liste noire s'il agit frauduleusement ou être puni d'une autre manière appropriée.

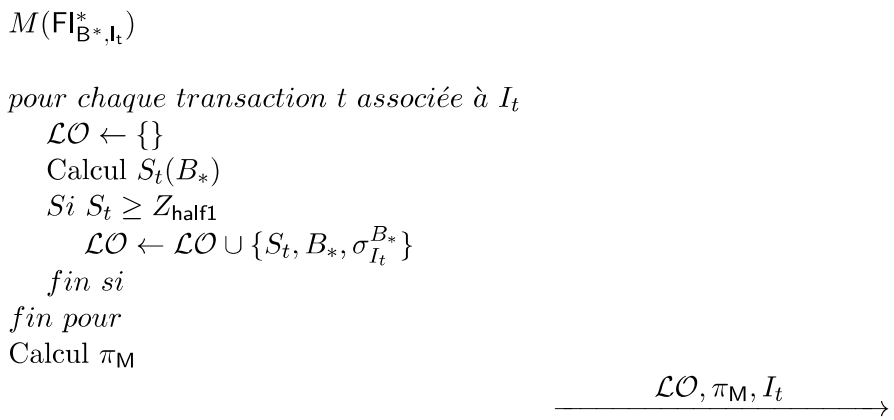

FIGURE 3.8.: Phase d'accusation de PIMENTO

Remarque 3.1. : Vie privée contre traçage de traîtres. La manière probabiliste dans laquelle nous exécutons le protocole OT^2_1 est construite pour préserver la vie privée de l'acheteur – en ne relevant pas plus que le strict nécessaire des indices de la marque contenue dans la copie du contenu récupérée par cet acheteur – et la capacité des codes

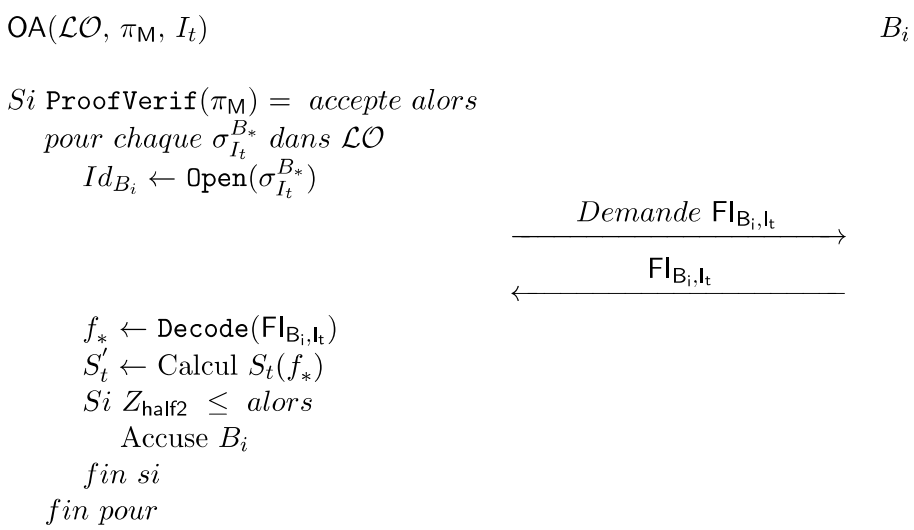

FIGURE 3.9.: Phase d'ouverture de PIMENTO.

de Tardos de tracer les traîtres – en donnant le *halfword* au vendeur. De plus, le vendeur ne doit pas apprendre si l'entrée reçue (c'est-à-dire, l'indice et la preuve vérifiable) est la vraie ou l'entrée simulée avant de la vérifier. Par conséquent, avec une probabilité non négligeable, le vendeur retrouvera un peu moins que la moitié exacte des indices (voir figure 3.12). De plus, la limite que nous donnons dans la section 3.4 ne résiste que si le vendeur récupère au moins la moitié de ces positions. De plus, il faut noter que si le vendeur reçoit moins de la moitié de la marque, la limite du traçage de traîtres est seulement marginalement dégradée. Cependant, pour la validité formelle des théorèmes de la section 3.4 nous faisons l'hypothèse que le vendeur retrouve toujours le *halfword* entier.

3.2.2. Assurer la non-chaînabilité des contenus

Assurer la non-chaînabilité des contenus n'est pas une propriété facile à réaliser, car le traçage de traîtres est assuré par les codes de Tardos en calculant des scores d'accusation, pour lesquels le contenu doit être connu. Notre solution consiste à ce que l'acheteur envoie à l'OA le contenu qu'il veut acquérir, ainsi qu'une signature de groupe sur les paramètres envoyés par le vendeur. Ceci permet à l'OA d'associer les signatures spécifiques avec les contenus. Quand le vendeur suspecte qu'une copie forgée est diffusée, il calcule le score d'accusation de chaque transaction, indépendamment du contenu acheté qu'il ne connaît pas. Tous les scores au-dessus du seuil d'accusation sont envoyés à l'OA ainsi que l'indice du contenu forgé, les signatures des transactions suspectes et la preuve que les calculs ont été fait correctement. Ensuite, l'OA peut supprimer les signatures qui ne correspondent pas au contenu considéré et ensuite révoquer l'anonymat des utilisateurs dont les scores d'accusations sont élevés et dont les vraies signatures sont associées à ce contenu.

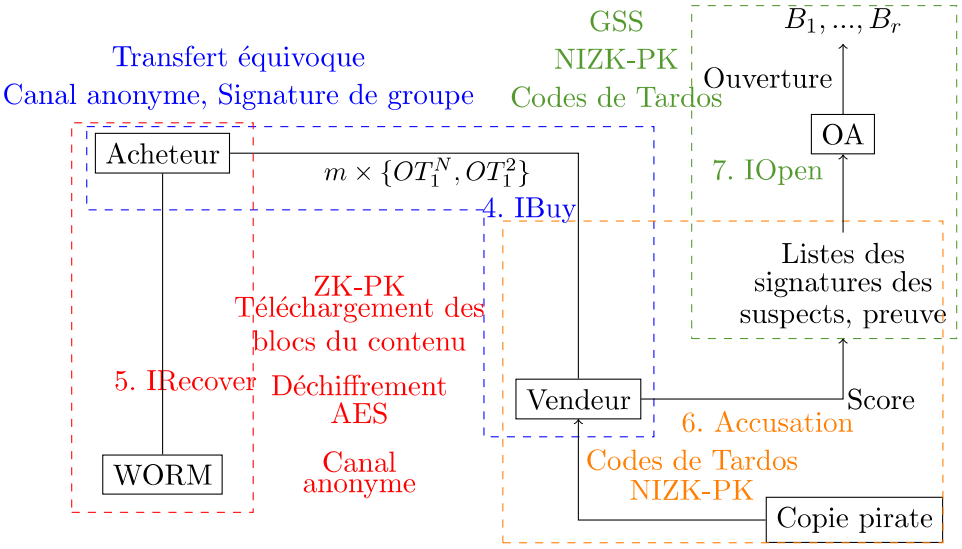

FIGURE 3.10.: Récupération d'un contenu et processus d'accusation

Nous décrivons ci-dessous la version modifiée de l'algorithme pour PIMENTO+, notre protocole assurant la non-chaînabilité des contenus.

`IBuy`. Le vendeur et les acheteurs communiquent *via* un canal de communication anonyme. Lorsqu'un acheteur B_i veut acquérir un contenu I_t auprès d'un vendeur, le vendeur commence par lui envoyer l'étiquette temporelle TS courante et un numéro de transaction t aléatoire. L'acheteur utilise sa clef privée de signature de groupe $\mathsf{sk}_{\mathsf{B_i}}$ pour signer ce message, composé de TS et de t, avant d'envoyer le message chiffré à l'OA ainsi que la signature notée $\sigma_{I_t}^{B_i}$ et l'identifiant I_t du contenu qu'il souhaite acquérir. La clef utilisée lors du chiffrement du message est la clef publique certifiée de l'OA. Ensuite, l'OA vérifie la validité des signatures. Si cette vérification échoue, il avorte le protocole et produit en sortie le symbole $\perp$. Sinon, l'OA stocke la paire $(\sigma_{I_t}^{B_i}, I_t)$ dans sa base de données $\mathcal{D}$ de transactions et retourne une version certifiée de la signature $\sigma_{I_t}^{B_i}$, qui est noté $\mathsf{Cert}(\sigma_{\mathsf{I_t}}^{\mathsf{B_i}})$. L'acheteur transmet cette signature certifiée au vendeur qui la vérifie. Si une de ces vérifications échoue, le vendeur avorte le protocole et produit en sortie le symbole $\perp$.

Ensuite, l'acheteur et le vendeur exécutent un protocole d'OT bruité. Plus précisément, le vendeur génère ℓmN valeurs aléatoires $R_t^{i,j}$, dans lesquelles $i \in \{1, \dots N\}, j \in \{1, \dots m\}$ et ℓ est le nombre total de contenus. L'entrée du vendeur est composée de toutes les matrices de clefs κ_{I_t} pour tous les contenus et de toutes les valeurs aléatoires. Premièrement, l'acheteur exécute un protocole de $OT_1^{N \cdot \ell}$, permettant de récupérer une clef $\mathsf{k}_{I_t}^{i,1}$ pour des $i \in \{1, \dots, N\}$ et les valeurs aléatoires correspondantes $R_t^{i,1}$. Ensuite, le vendeur exécute un protocole de OT_1^2 avec l'acheteur, dans lequel les entrées de l'acheteur sont (1) un tuple $(i, R_t^{i,1}, \pi_{\mathsf{i}}, \sigma_i)$, tel que π_{i} est une preuve de connaissance NIZK que l'acheteur connaît une clef $\mathsf{k}_{I_t}^{i,1}$ et σ_i est une signature

sur la preuve concaténée avec la valeur aléatoire $R_t^{i,1}$, et (2) un tuple (j, r_j), avec les deux valeurs choisies aléatoirement. Dans ce cas, la preuve NIZK de connaissance est une preuve d'appartenance à un ensemble. Dans cette preuve, le prouveur (ici l'acheteur) a un témoin consistant en la clef $\mathsf{k}_{I_t}^{i,1}$, qui doit appartenir à l'ensemble $\{\mathsf{k}_{I_t}^{1,1} \dots \mathsf{k}_{I_t}^{N,1}\}$ de clefs légitimes possibles pour le premier bloc du contenu t. Le vendeur reçoit le tuple et vérifie la validité de la preuve et de la signature. Si la preuve est valide alors le vendeur stocke l'indice i dans le *halfword* Hw. Sinon, le vendeur et l'acheteur exécutent une deuxième fois un $OT_1^{N \cdot \ell}$ pour récupérer une seconde clef secrète. Le processus continue jusqu'à ce que m clefs soient récupérées par l'acheteur. De plus, le vendeur vérifie à intervalles réguliers, dont la fréquence dépend de m, qu'il récupère au moins la moitié des indices pour lesquelles l'acheteur a obtenu des clefs secrètes. Sinon, le vendeur avorte la transaction. Vu que la taille de m est présumée grande, il est raisonnable de s'attendre à ce que le nombre d'indices dans le *halfword* soit très proche de la moitié des clefs que l'acheteur a récupérées. À la fin de cette transaction, l'acheteur a récupéré un ensemble $\mathbf{KI}_{B,I_t}$ de taille m dont les éléments sont des tuples de la forme $(i, \mathsf{k}_I^{i,j})_{j=1}^m$. De plus, le vendeur a récupéré un ensemble de Hw d'indices.

`IRecover`. L'acheteur ayant des ensembles clefs secrètes et indices, peut récupérer le contenu personnalisé $\mathsf{FI}_{\mathsf{B},\mathsf{I_t}}$. En plus de récupérer ce contenu personnalisé, l'acheteur génère aussi une preuve de connaissance NIZK qu'il connaît une version personnalisée du contenu I_t et l'envoie avec la signature de groupe *via* un canal de communication anonyme et sécurisé à OA [27]. OA ajoute la preuve aux tuples de la base contenant les signatures de groupe et les contenus.

`Accuse`. Pour un contenu $\mathsf{FI}^*_{\mathsf{B}^*,\mathsf{I}}$ d'un contenu I, le vendeur calcule les scores partiels des codes de Tardos S_t pour chaque enregistrement de transaction t et les compare avec le seuil d'accusation partielle Z_{half1}. Pour chaque score plus élevé que le seuil, le vendeur ajoute à la liste $\mathcal{LO}$ la signature $\sigma_{I_t}^{B*}$ reçue durant la transaction. Ensuite, une fois les calculs effectués, le vendeur génère aussi une preuve de connaissance NIZK que les scores d'accusation ont été correctement calculés. Cette NIZK de conformité a un témoin composé de la transcription des transactions des acheteurs suspects, leurs scores ainsi que le seuil Z_{half1}. Les scores sont calculés comme dans [128]. L'énoncé prouvé par le témoin est que les scores sont correctement calculés, qu'ils sont plus grands que le seuil Z_{half1} et que les messages signés et envoyés avec la preuve sont bien ceux associés aux transactions. Finalement, le vendeur transmet l'indice du contenu forgé I_t, la liste des signatures $\mathcal{LO}$ et la preuve π_{M}.

`Open`. À partir de l'indice de la copie forgée, la liste de signatures et la preuve de conformité des calculs, l'OA vérifie la preuve. Si celle-ci est vérifiée, pour chaque signature $\sigma_{I_t}^B$, si $t' \neq t$ et que l'OA a une preuve valide fournie par l'acheteur qu'il a bien acheté le contenu I_t, l'OA supprime la signature de la liste $\mathcal{LO}$. Finalement, l'OA ouvre les signatures restantes avec la clef maître permettant de lever l'anonymat. Ces

27. Pour la preuve de connaissance NIZK, le témoin est la séquence de clefs et les bits récupérés $\mathsf{k}_{I_t}^{i,1}, \mathsf{W}(I_t^i, f_{I_t}^{i,j})$ correspondant à le contenu I_t. L'énoncé prouvé est que cette séquence de bits est la même qu'une séquence de bits de marquage chiffrés et stockés dans le WORM du contenu I_t et qu'il y a exactement une entrée pour chaque $i \in \{1 \dots m\}$.

identités ne sont par contre pas révélées au vendeur.

3.3. Modèle de sécurité

Dans cette section, nous commençons par décrire le modèle d'adversaire de manière formelle. Ensuite, nous définissons les exigences de sécurité et de vie privée pour les protocoles de personnalisation de contenus asymétrique préservant la vie privée.

Modèle d'adversaire. Chaque contenu du vendeur est divisé en blocs de taille égale (la taille dépend du média et du schéma d'insertion W). Dans ce travail, nous supposons que les contenus ont tous m blocs notés $I^1, \ldots, I^m$. Si un contenu est composé de moins de blocs, le vendeur fait du rembourrage (complète) le contenu avec des blocs additionnels.

Nous définissons les propriétés de sécurité et de vie privée d'un protocole de personnalisation de contenus préservant la vie privée avec traçage de traître (`PFP-TT`, *Privacy-preserving Fingerprinting Protocol with Traitor-Tracing*). Intuitivement, un protocole `PFP-TT` est constitué des algorithmes suivants :

- Un algorithme d'*initialisation* (`Setup`), qui fournit un ensemble de paramètres publics et privés.
- Un algorithme d'*enregistrement d'acheteurs* (`BReg`) grâce auquel un acheteur s'enregistre auprès de CA et reçoit des accréditations privées de groupe.
- Un algorithme de *préparation de contenus* (`IPrep`), permettant au vendeur de préparer les contenus τ_{I} et une preuve π_{I} que les contenus ont été correctement préparés.
- Un algorithme *achat* (`IBuy`), permettant à l'acheteur de récupérer un ensemble de clefs et de positions indiquant les entrées dans la matrice τ_{I} dans laquelle l'acheteur récupère le contenu. Le vendeur reçoit un *halfword*, lui permettant de détecter des acheteurs potentiellement malveillants.
- Un algorithme *Récupération* (`IRecover`), permettant à l'acheteur de récupérer un contenu marqué avec une empreinte unique.
- Un algorithme d'*accusation* (`Accuse`) permettant au vendeur de générer une liste d'identifiants d'acheteurs et une preuve que ces acheteurs sont malveillants.
- Un algorithme d'*ouverture* (`Open`), par lequel OA lève l'anonymat des acheteurs malveillants en se basant sur une preuve de leur culpabilité.

Définitions des algorithmes. Plus formellement, les capacités d'un `PFP-TT` sont définies comme le tuple d'algorithmes suivant : `PFP-TT` = (`Setup`, `BReg`, `IPrep`, `IBuy`, `IRecover`, `Open`).

`Setup` : Quand on fournit en entrée le paramètre de sécurité 1^λ, cet algorithme retourne les paramètres secrets spar (qui sont répartis entre CA et OA) et les paramètres publics ppar qui sont connus de toutes les parties. Nous supposons que les algorithmes restants prennent tous en entrée les paramètres publics ppar.

`BReg` : Quand on donne comme entrée spar et l'identité d'un acheteur B, l'algorithme d'enregistrement des acheteurs fournit en sortie une clef secrète sk_{B} pour B ou

$\perp$.

`IPrep` : Quand on donne un contenu I en entrée, l'algorithme de préparation du contenu sort le contenu τ_{I} (dans notre cas un WORM [98]), une preuve π_{I} que le WORM de ce contenu est correctement formé, une matrice de clef κ_{I_I} et une matrice $\mathcal{F}_I$ contenant les bits servant à personnaliser le contenu.

`IBuy` : Cet algorithme interactif acheteur-vendeur prend en entrée I, la clef secrète sk_{B} de l'acheteur et une matrice de clefs κ_{I_I} générée lors de la préparation du contenu. La sortie est composée d'un ensemble de clefs $\mathbf{KI}_{B,I}$ et de diverses informations auxiliaires $\mathsf{aux}_{\mathsf{B,I}}$ (dans notre cas le *halfword*).

`IRecover` : Quand on donne en entrée l'ensemble des clefs $\mathbf{KI}_{B,I}$, le contenu préparé τ_{I} et la preuve π_{I}, l'algorithme de récupération retourne un contenu marqué $\mathsf{FI}_{\mathsf{B,I}}$ ou le symbole $\perp$.

`Open` : Quand une preuve π_{V}, générée par le vendeur est fournie en entrée ainsi que spar, l'algorithme retourne un ensemble d'identités d'acheteurs, noté $\{B_i\}_{i=1}^{d}$ ou un symbole d'erreur $\perp$. La valeur maximum de d est le nombre d'acheteurs N exécutant une attaque par collusion.

`Accuse` : L'algorithme d'accusation prend en entrée une copie marquée FI et un ensemble de toutes les informations auxiliaires $\mathsf{aux}_{\mathsf{B,I}}$ obtenues lors d'une transaction honnête, et retourne une preuve π qu'un acheteur fait partie de la collusion.

Oracles formels. Pour le modèle de sécurité et de vie privée, nous considérons un adversaire $\mathcal{A}$ qui a accès aux oracles suivants pour qu'il interagisse avec eux comme s'il était une partie honnête. Ces oracles sont formellement présentés ci-dessous.

$\mathsf{Buy}^*(I, B, \mathsf{input}^*)$: Cet oracle autorise un adversaire (en particulier un vendeur malveillant) à dévier du protocole et exécuter l'algorithme `IBuy` pour un item I et un acheteur B avec l'entrée malveillante input^*. L'oracle retourne la sortie complète de l'algorithme `IBuy` et la transcription de la transaction. Intuitivement, cet oracle est utilisé par un vendeur malveillant qui essaie de briser la vie privée des utilisateurs.

$\mathsf{Execute}(I, B)$: Cet oracle prend en entrée l'identifiant d'un item I et l'identifiant d'un acheteur B et simule l'exécution de l'algorithme `IBuy` pour un acheteur B et un item I pour une entrée honnête du vendeur. L'oracle retourne deux valeurs produites par l'algorithme d'achat : la clef $\mathbf{KI}_{B,I}$ et l'information auxiliaire $\mathsf{aux}_{\mathsf{B,I}}$, ainsi que la transcription de la transaction. Contrairement à l'oracle précédent, dans lequel le vendeur peut dévier du protocole, l'interaction entre acheteur et vendeur est honnête et exécutée pour une entrée donnée. Cet oracle permet aux adversaires de démarrer une interaction honnête.

$\mathsf{BBuy}(I, \mathsf{sk}_{\mathsf{B}})$: Cet oracle prend en entrée la clef secrète de l'acheteur sk_{B} ainsi qu'un item I et exécute l'algorithme `IBuy`, retournant $\mathbf{KI}_{B,I}$ et la transcription de la transaction. Dans ce troisième oracle, l'adversaire est supposé connaître aussi la clef secrète d'un utilisateur spécifique (par exemple, cette clef peut être obtenue d'un acheteur corrompu ou si un acheteur fait partie de la collusion de traîtres.)

$\mathsf{Open}(\pi_{\mathsf{M}})$: Cet oracle prend en entrée la preuve π_{M} et exécute l'algorithme d'ouverture `Open` sur la preuve π_{M} et les paramètres secrets spar, fournissant en sortie un ensemble d'identités $\{B_i\}_{i=1}^{d}$. L'oracle `Open` retourne cet ensemble d'identités. Cet oracle permet à l'adversaire d'apprendre le résultat de la requête d'ouverture venant du juge en cas de mauvaise conduite.

$\mathsf{Corrupt}(B)$: Cet oracle prend comme entrée l'identifiant d'un acheteur B et donne en sortie la clef secrète de l'acheteur sk_{B}.

$\mathsf{Collude}(\{\mathsf{FI}_{\mathsf{B_i},\mathsf{I}}\}_{i=1}^{k}, \mathsf{strategy})$: Cet oracle prend en entrée un ensemble d'au plus $k \leq c$ copies marquées qui ont été légitimement achetées $\{\mathsf{FI}_{\mathsf{B_i},\mathsf{I}}\}_{i=1}^{k}$ et une stratégie $\mathsf{strategy}$ pour forger une copie marquée $\mathsf{FI}_{\tilde{\mathsf{B}},\tilde{\mathsf{I}}}$. Cette stratégie peut être arbitraire avec l'exception de la restriction suivante : si, pour des blocs i d'un item, le bloc marqué $\mathsf{fb}_I^{i,j}$, récupéré par *tous* les colluders, possède le bit de l'empreinte $f_I^{i,j}$, alors le bloc marqué correspondant de l'item forgé $\mathsf{FI}_{\mathsf{B}^*,\mathsf{I}^*}$ *doit* avoir le bit d'empreinte $f_I^{i,j}$. Cette restriction est une conséquence directe de la marking assumption décrite précédemment. Intuitivement, cet oracle simule toutes les stratégies que peuvent faire les membres d'une collusion.

$\mathsf{Accuse}(\mathsf{FI},)$: Cet oracle exécute `Accuse`, quand l'entrée donnée est une copie marquée FI, une matrice de clefs κ_{I_I} et une matrice d'empreinte $\mathcal{F}_I$, fournissant en sortie la preuve π. Cet oracle simule le processus d'accusation du côté du vendeur pour un item donné.

BReg : Quand l'identité d'un acheteur B est donnée en entrée, cet oracle exécute l'algorithme d'enregistrement de l'acheteur, fournissant à B une clef secrète sk_{B} et certifiant B comme acheteur légitime.

3.3.1. Exigences de sécurité et de respect de la vie privée de PIMENTO

Dans cette section, nous présentons les exigences de sécurité et de vie privée que notre protocole `PFP-TT` doit garantir. Plus précisément, nous définissons le jeu contre la sécurité pour chaque propriété en fonction d'un oracle additionnel appelé Test, qui change à chaque jeu pour refléter les exigences des propriétés de sécurité et de vie privée. Dans chaque jeu de sécurité, nous donnons accès à l'adversaire à un sous-ensemble des algorithmes et des oracles présentés dans la section précédente aussi bien qu'à l'oracle Test.

Conformité. Dans ce jeu, le challenger exécute en premier l'algorithme `Setup`, qui donne en sortie les paramètres publics (ppar) et secrets (spar) avant de procéder à la préparation de tous les contenus I en utilisant l'algorithme `IPrep`. L'adversaire possède ppar et la preuve des tuples correspondants $(\tau_{\mathsf{I}}, \pi_{\mathsf{I}})$ pour chaque contenu I. L'adversaire peut, adaptativement, enregistrer des acheteurs et ensuite interroger l'oracle $\mathsf{Test}^{\mathsf{Corr}}$, qui lance une exécution acheteur-vendeur honnête, avant d'appeler l'algorithme de récupération. Cet oracle retourne 0 si l'exécution honnête est mal formée (c'est-à-dire, l'item en entrée ou l'identifiant de l'acheteur correspond à une valeur qui n'existe pas) ou s' il échoue à retrouver l'ensemble correct des blocs marqués. Sinon, l'oracle retourne 1. Plus formellement, nous définissons Test comme suit :

— $\mathsf{Test}^{\mathsf{Corr}}$: Quand l'entrée fournie est l'identifiant d'un item I ainsi que l'identifiant d'un acheteur B, $\mathsf{Test}^{\mathsf{Corr}}$ exécute $\mathsf{Execute}(I, B)$, et sort les clefs $\mathbf{KI}_{B,I} = \{(j, \widetilde{\mathsf{k}}_I^{i,j})\}_{j\in\{1,\ldots,N\}}$, pour des valeurs consécutives de i (si ces valeurs ne sont pas consécutives ou si elles sont dans un mauvais format $\mathsf{Test}^{\mathsf{Corr}}$ retourne 0). L'algorithme `IRecover` est par la suite lancé en fournissant en entrées les clefs $\mathbf{KI}_{B,I}$, la table τ_{I} et la preuve π_{I}. La sortie est la série de blocs $\mathsf{FI}_{\mathsf{B},\mathsf{I}}$ (sinon si $\perp$ est en sortie, l'oracle $\mathsf{Test}^{\mathsf{Corr}}$ retourne 0). L'oracle teste si pour chaque entrée $[\mathsf{FI}_{\mathsf{B},\mathsf{I}}]_{i,j}$, il s'avère que $[\tau_{\mathsf{I}}]_{i,j} = \mathsf{P}([\mathsf{FI}_{\mathsf{B},\mathsf{I}}]_{i,j}, [\kappa_{I_I}]_{i,j})$ pour des fonctions P de préparation de trappe à sens unique. Si cette dernière vérification échoue, l'oracle produit en sortie 0, sinon il produit 1.

L'adversaire gagne si au moins une requête $\mathsf{Test}^{\mathsf{Corr}}$ retourne 0. Nous définissons l'*avantage* de l'adversaire $\mathcal{A}$ comme $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{correct}} = \mathbb{P}[\mathcal{A}\ \mathsf{wins}]$.

Définition 3.1 (Conformité)**.** *Un protocole* `PFP-TT` *est* $(N_{\mathsf{Test}}, \epsilon)$*-conforme si tout adversaire* $\mathcal{A}$*, opérant en temps polynomial contre la conformité de* `PFP-TT` *faisant au plus* N_{Test} *requête à l'oracle* $\mathsf{Test}^{\mathsf{Corr}}$*, gagne avec un avantage* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{correct}} \leq \epsilon$*. Asymptotiquement, le protocole est* N_{Test}*-conforme si tout adversaire* $\mathcal{A}$ *a une probabilité négligeable de gagner avec au plus* N_{Test} *requêtes.*

Exigence de sécurité

Anti-framing. Cette propriété garantit l'anti-framing (c'est-à-dire, qu'un acheteur honnête ne peut pas être accusé par un vendeur malveillant) et la disculpation (c'est-à-dire, qu'un acheteur honnête ne peut pas être accusé par une collusion d'acheteurs malveillants) en considérant une collusion générique entre un vendeur malveillant et un ensemble d'acheteurs malveillants. L'adversaire utilise l'information secrète du vendeur et interroge $\mathsf{Corrupt}$ pour faire une collusion composée d'acheteurs. Il peut aussi interroger $\mathsf{Collude}$ (pour n'importe lequel des acheteurs corrompus) et un oracle de test, qui autorise le vendeur à simuler l'algorithme d'ouverture. L'objectif de $\mathsf{Test}^{\mathsf{NoFrame}}$ est de vérifier si $\mathcal{A}$ peut produire une preuve convaincante permettant de lever l'anonymat d'un acheteur honnête (c'est-à-dire, non corrompu) quand l'oracle est interrogé. L'adversaire gagne si au moins une interrogation de $\mathsf{Test}^{\mathsf{NoFrame}}$ retourne 1. Plus formellement, $\mathsf{Test}^{\mathsf{NoFrame}}$ se comporte comme suit :

— $\mathsf{Test}^{\mathsf{NoFrame}}$: Quand une preuve π_{M} est fournie en entrée, $\mathsf{Test}^{\mathsf{NoFrame}}$ lance l'oracle Open comme une boîte noire et reçoit l'ensemble des identités $\{B_i\}_{i=1}^{d}$. L'oracle vérifie si au moins une identité fournie en sortie par Open est non corrompue au moment de l'interrogation de $\mathsf{Test}^{\mathsf{NoFrame}}$. Si l'état est vrai, l'oracle sort 1 sinon 0.

Nous définissons l'avantage de l'adversaire comme $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{no-frame}} = \mathbb{P}[\mathcal{A}\ \mathsf{wins}]$.

Définition 3.2 (Anti-framing)**.** *Un protocole* `PFP-TT` *est* $(N_{\mathsf{Test}}, \epsilon)$*-unframeable si tout adversaire* $\mathcal{A}$*, opérant en temps polynomial contre l'anti-framing de* `PFP-TT` *en faisant au plus* N_{Test} *requête à l'oracle* $\mathsf{Test}^{\mathsf{NoFrame}}$*, gagne avec un avantage* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{no-frame}} \leq \epsilon$*. Asymptotiquement, le protocole est* N_{Test}*-unframeable s'il est* $(N_{\mathsf{Test}}, \nu(1^{\lambda}))$*-unframeable.*

Traçage de Traîtres. En traçage de traîtres, un adversaire peut utiliser les paramètres publics et toutes les clefs privées des acheteurs qu'il contrôle pour simuler des transactions et obtenir une copie marquée légitime $\mathsf{FI}_{\mathsf{B},\mathsf{I}}$. L'oracle $\mathsf{Test}^{\mathsf{TT}}$ exécute l'oracle $\mathsf{Collude}$ sur tous les sous-ensembles des copies légitimes, avant d'appeler les oracles Accuse et Open (simulant ainsi la tentative d'un vendeur de tracer une copie pirate), et il gagne si l'ouverture ne révèle pas d'identité associée avec une copie légitime dans le sous-ensemble original. Cette propriété peut être plus formellement décrite de la manière suivante :

— $\mathsf{Test}^{\mathsf{TT}}$: Quand un ensemble de copies marquées légitimes $\{\mathsf{FI}_{\mathsf{B_i},\mathsf{I}}\}_{i=1}^{k}$ et une stratégie $\mathsf{strategy}$ sont fournis en entrée, $\mathsf{Test}^{\mathsf{TT}}$ exécute $\mathsf{Collude}$, donnant en sortie une copie forgée $\mathsf{FI}_{,}$. Par la suite, il exécute Accuse avec l'entrée $\mathsf{FI}_{,}$ et reçoit la preuve π. Cette preuve est donnée comme entrée à l'oracle Open, qui retourne un ensemble d'identité $\{B_j\}_{j=1}^{d}$. S'il existe des acheteurs B^* tels qu'une des entrées était $\mathsf{FI}_{\mathsf{B}^*,\mathsf{I}}$ *et* que B^* est parmi les sorties de l'interrogation de Open, alors l'oracle $\mathsf{Test}^{\mathsf{TT}}$ retourne 1 et la preuve π sinon 0.

L'avantage de l'adversaire est $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{TT}} = \mathbb{P}[\mathcal{A}\ \mathsf{wins}]$.

Définition 3.3 (Traçage de Traîtres)**.** *Un protocole* `PFP-TT` *est* $(N_{\mathsf{Test}}, c, \epsilon)$*-traçage-de-traître si tout adversaire* $\mathcal{A}$*, opérant en temps polynomial contre la propriété de traçage de traîtres de* `PFP-TT` *en faisant au plus* N_{Test} *requêtes à l'oracle* Test^{TT} *pour au plus* c *colluders, gagne avec un avantage* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{TT}} \leq \epsilon$*. Asymptotiquement, le protocole est* N_{Test}*-traceur-de-traître s'il est* $(N_{\mathsf{Test}}, \nu(1^{\lambda}))$*-traçage-de-traître.*

Exigence pour la protection de la vie privée

Non chaînabilité des acheteurs. Suivant les approches de Hermans, Pashalidis, Vercauteren et Preneel [69] et Gambs, Onete et Robert [57], ce jeu considère que l'adversaire est un vendeur, qui peut être honnête-mais-curieux (hbc) ou malveillant (mal). Ce comportement est noté par un drapeau $\mathsf{flag} \in \{\mathsf{hbc}, \mathsf{mal}\}$. Ainsi, $\mathcal{A}$ reçoit les paramètres publics et le vendeur partage spar, avant de lancer `IPrep` à volonté. Si le drapeau $\mathsf{flag} = \mathsf{hbc}$, l'adversaire peut interroger l'oracle $\mathsf{Execute}$ et $\mathsf{Corrupt}$, tandis que si $\mathsf{flag} = \mathsf{mal}$, $\mathcal{A}$ peut interroger Buy^* et $\mathsf{Corrupt}$. L'oracle adaptatif $\mathsf{Test}^{\mathsf{BUnlink}}$ prend en entrée deux acheteurs B_i et B_j, en choisissant toujours le premier ou le second, en fonction d'un bit caché b. L'adversaire peut lancer $\mathsf{Execute}$ (respectivement Buy^*) à volonté avec l'acheteur avant, éventuellement, de libérer les deux acheteurs. Plus formellement, l'oracle Test se comporte comme suit :

— $\mathsf{Test}^{\mathsf{BUnlink}}$: quand deux identités d'acheteurs, B_i et B_j, sont passées en entrée ainsi que le paramètre $\mathsf{text} \in \{\mathsf{draw}, \mathsf{free}\}$, l'oracle $\mathsf{Test}_b^{\mathsf{BUnlink}}$, qui garde une base de données interne $\mathcal{D}_{\mathsf{Test}^{\mathsf{BUnlink}}}$, en associant toujours la première ou la seconde identité à une poignée handle dépendant du bit d'entrée b. Dans ce mode, une fois que la requête $\mathsf{Test}^{\mathsf{BUnlink}}(\cdot, \cdot, \mathsf{draw})$ est exécutée, l'adversaire peut interagir avec l'acheteur anonymisé au moyen de, respectivement, l'oracle $\mathsf{Execute}$ et Buy^* (nous avons modifié ces oracles pour prendre en entrée la poignée handle au lieu de l'identifiant de l'acheteur). L'adversaire peut aussi choisir d'interagir avec les autres acheteurs ou de les corrompre. Finalement, l'adversaire libère les deux

acheteurs au moyen de la requête $\mathsf{Test}^{\mathsf{BUnlink}}(\cdot,\cdot,\mathsf{free})$. Si l'adversaire interroge l'oracle Test avec, comme entrée draw tandis que la poignée actuelle n'a pas été libérée, cet oracle retourne $\perp$. Similairement, essayer de libérer la poignée alors qu'aucune poignée n'est associée à un acheteur donnera la sortie $\perp$.

L'adversaire gagne s'il peut deviner le bit b. Nous considérons les mêmes classes d'adversaires que celles définies par Vaudenay [127] : l'adversaire *faible* ne peut pas corrompre, l'adversaire *hardi* peut seulement suivre les requêtes de corruption par d'autres requêtes de corruption et l'adversaire *fort* peut tricher arbitrairement sans aucune restriction. Nous définissons l'avantage de l'adversaire de ce jeu $\mathcal{A}$ comme $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{ID-priv}} := \mathbb{P}[\mathcal{A}\ \mathsf{wins}] - \frac{1}{2}$.

Définition 3.4 (Non chaînabilité des acheteurs)**.** *Un protocole* PFP-TT *est* $(N_{\mathsf{Test}}, \epsilon)$-Type-*acheteur-non-chainable avec le respect de* flag-*vendeur (dans lequel* $\mathsf{flag} \in \{\mathsf{hbc}, \mathsf{mal}\}$*) si tout adversaire* $\mathcal{A}$*, opérant en temps polynomial, jouant* Type-*acheteur non-chaînable de* PFP-TT *(pour* $\mathsf{Type} \in \{$ *weak, forward, strong*$\}$*) faisant au plus* N_{Test} *requête à l'oracle* $\mathsf{Test}^{\mathit{BUnlink}}$*, gagne avec un avantage* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{ID-priv}} \leq \epsilon$*. Asymptotiquement, le protocole est* N_{Test}*-non-chaînable-au-niveau-des-acheteurs s'il est* $(N_{\mathsf{Test}}, \nu(1^{\lambda}))$*-non-chaînable-au-niveau-des-acheteurs.*

Non-chaînabilité des contenus. Cette propriété exige que le vendeur malveillant ne puisse apprendre quel contenu un acheteur a acheté. Nous définissons cette propriété comme un jeu d'indistinguabilité gauche-ou-droite, avec un vendeur jouant le rôle d'adversaire, commençant par déclarer le drapeau honnête-mais-curieux (hbc) ou malveillant (mal) $\mathsf{flag} \in \{\mathsf{hbc}, \mathsf{mal}\}$. L'adversaire peut interroger $\mathsf{Corrupt}$ et $\mathsf{Execute}$ ou Buy^*. L'oracle $\mathsf{Test}^{\mathsf{lUnlink}}$ correspondant permet à l'adversaire de lancer une exécution avec un acheteur choisi pour l'un des deux contenus. Dans cette configuration, l'objectif de l'adversaire est de deviner le bit b gauche-ou-droite. Cette propriété garantit aussi la non-chainabilité entre deux achats effectués par le même acheteur (supposant que chaque achat est effectué lors de transactions différentes). Plus formellement, l'oracle $\mathsf{Test}^{\mathsf{lUnlink}}$ se comporte comme suit :

— $\mathsf{Test}^{\mathsf{lUnlink}}$: Quand l'identité d'un acheteur B, deux identifiants de contenu I_1 et I_2 sont fournis en entrée et l'entrée du vendeur input^*, si $\mathsf{flag} = \mathsf{hbc}$, $\mathsf{Test}_b^{\mathsf{lUnlink}}$ ignore la dernière entrée et exécute $\mathsf{Execute}$ sur l'entrée une ou deux (en fonction du bit b), pour l'acheteur B. Sinon, il utilise Buy^* pour l'entrée B, input^* et l'item choisi.

L'avantage de l'adversaire est défini comme $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{contenu-priv}} := \mathbb{P}[\mathcal{A}\ \mathsf{wins}] - \frac{1}{2}$.

Définition 3.5 (Non-chaînabilité des contenus)**.** *Un protocole* PFP-TT *est* $(N_{\mathsf{Test}}, \epsilon)$*-contenu-non-chainable avec le respect de* flag*-vendeur (dans lequel* $\mathsf{flag} \in \{\mathsf{hbc}, \mathsf{mal}\}$*) si tout adversaire* $\mathcal{A}$*, opérant en temps polynomial contre la non-chainabilité des contenus de* PFP-TT *faisant au plus* N_{Test}*requêtes à l'oracle* $\mathsf{Test}^{\mathit{lUnlink}}$*, gagne avec un avantage* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{contenu-priv}} \leq \epsilon$*. Asymptotiquement, le protocole est* N_{Test}*-non-chaînable-au-niveau-des-contenus s'il est* $(N_{\mathsf{Test}}, \nu(1^{\lambda}))$*-non-chaînable-au-niveau-des-contenus.*

3.4. Analyse de sécurité et de respect de la vie privée

Le théorème suivant caractérise les propriétés de sécurité et de vie privée fournies par PIMENTO (section 3.2.1). Nous incluons aussi la propriété de la non-chaînabilité de PIMENTO+ (section 3.2.2).

Théorème 3.4.1 (Securité de PIMENTO). *PIMENTO est implémenté avec un schéma de signature de groupe* $\texttt{GSScheme} = (\texttt{GSKGen}, \texttt{Join}, \texttt{Sign}, \texttt{Vf}, \texttt{Open}, \texttt{Revoke})$*, un schéma de chiffrement symétrique* $\texttt{EScheme} = (\texttt{EKGen}, \texttt{Enc}, \texttt{Dec})$*, un schéma de certification* $\texttt{Cert} = (\texttt{CSign}, \texttt{CVf})$*, et protocole de transfert équivoque* 1*-parmi-*$N \cdot \ell$*. Nous supposons qu'il existe un schéma de chiffrement à clef publique IND-CCA-secure (pour* OA*), un canal de communication d'un côté anonyme et d'un côté authentifié entre le vendeur et les acheteurs, ainsi que l'existence de deux preuves de connaissance à divulgation nulle de connaissance non interactive* $\mathsf{NIZK\text{-}PK}_1$ *et* $\mathsf{NIZK\text{-}PK}_2$ *(pour prouver respectivement (1) que les matrices des contenus sont bien formées et (2) la conformité du calcul des scores d'accusation). Dans ce cas, PIMENTO fournit les propriétés suivantes :*

Conformité. Pour chaque adversaire $\mathcal{A}$ $(N_{\mathsf{Test}}, \epsilon)$*-conforme, il existe : un adversaire* $\mathcal{A}_1$ *contre la conformité de* GSScheme*, un adversaire* $\mathcal{A}_2$ *contre la conformité du schéma de* Cert *pour* CA*, un adversaire* $\mathcal{A}_3$ *contre la conformité du schéma NIZK-PK* $\mathsf{NIZK\text{-}PK}_1$*, un adversaire* $\mathcal{A}_4$ *contre la conformité du schéma de chiffrement symétrique et un adversaire* $\mathcal{A}_5$ *contre la conformité du schéma de transfert équivoque, tel que*

$$\epsilon \leq \sum_{i=1}^{3} \mathsf{Adv}_{\mathcal{A}_i}^{\mathsf{conformite}} + c\ell m \mathsf{Adv}_{\mathcal{A}_4}^{\mathsf{conformite}} + mN_{\mathsf{Test}} \mathsf{Adv}_{\mathcal{A}_5}^{\mathsf{conformite}}.$$

Non-chaînabilité des acheteurs. Pour tout adversaire $\mathcal{A}$ $(N_{\mathsf{Test}}, \epsilon)$*-non-chaînabilité-des-acheteurs contre notre protocole, dans lequel un total de* n *acheteurs sont enregistrés, il existe : un adversaire* $\mathcal{A}_1$ *contre l'anonymat total de* GSScheme*, un adversaire* $\mathcal{A}_2$ *contre l'anonymat du canal de communication entre le marchand et les acheteurs, un adversaire* $\mathcal{A}_3$ *contre la sécurité du canal entre* CA *et les acheteurs, un adversaire* $\mathcal{A}_4$ *contre la solidité de la preuve de connaissance NIZK* $\mathsf{NIZK\text{-}PK}_1$*, tel que :*

$$\epsilon \leq N_{\mathsf{Test}} \mathsf{Adv}_{\mathcal{A}_1}^{\mathsf{full-anon}} + N_{\mathsf{Test}} \mathsf{Adv}_{\mathcal{A}_2}^{\mathsf{anon}} + n\mathsf{Adv}_{\mathcal{A}_3}^{\mathsf{sec}} + \ell^2 \mathsf{Adv}_{\mathcal{A}_4}^{\mathsf{snd}}.$$

Anti-framing. Pour tout adversaire $\mathcal{A}$ $(N_{\mathsf{Test}}, \epsilon)$*-anti-framing contre PIMENTO, il existe : un adversaire* $\mathcal{A}_1$ *contre la traçabilité totale de* GSScheme *et deux adversaires* $\mathcal{A}_2$ *et* $\mathcal{A}_3$ *contre la sûreté de, respectivement,* $\mathsf{NIZK\text{-}PK}_1$ *et* $\mathsf{NIZK\text{-}PK}_2$*, tel que :*

$$\epsilon \leq \mathsf{Adv}_{\mathcal{A}_1}^{\mathsf{full-trace}} + \ell^2 \mathsf{Adv}_{\mathcal{A}_2}^{\mathsf{Snd}} + N_{\mathsf{Test}} n \mathsf{Adv}_{\mathcal{A}_3}^{\mathsf{Snd}} + N_{\mathsf{Test}} n\delta.$$

Traçage de traîtres. Pour tout adversaire $\mathcal{A}$ $(N_{\mathsf{Test}}, c, \epsilon)$*-traçage-de-traître contre PIMENTO, il existe : un adversaire* $\mathcal{A}_1$ *contre la traçabilité totale de* GSScheme *et*

un adversaire $\mathcal{A}_2$ contre la sûreté de $\mathsf{NIZK\text{-}PK}_2$ *tel que :*

$$\epsilon \leq \mathsf{Adv}^{\mathsf{full-trace}}_{\mathcal{A}_1} + N_{\mathsf{Test}}\mathsf{Adv}^{\mathsf{Snd}}_{\mathcal{A}_2}.$$

Non-chaînabilité des contenus. Pour **PIMENTO+** *nous ajoutons la supposition qu'il existe* $\mathsf{NIZK\text{-}PK}_3$ *et* $\mathsf{NIZK\text{-}PK}_4$ *pour prouver respectivement (3) la connaissance d'une des valeurs $\ell \cdot N$ et (4) la possession d'une copie marquée d'un contenu ℓ. Pour tout adversaire $\mathcal{A}$ $(N_{\mathsf{Test}}, \epsilon)$-non-chainabilité-des-contenus contre* **PIMENTO+***(Section 3.2.2), il existe : un adversaire $\mathcal{A}_1$ contre la sécurité IND-CCA du schéma de chiffrement à clefs publiques pour* OA*, un adversaire $\mathcal{A}_2$ contre la sécurité du protocole de $OT_1^{\ell \times N}$, deux adversaires $\mathcal{A}_3$ et $\mathcal{A}_4$ contre les schémas de preuve à divulgation nulle de connaissance non interactive* $\mathsf{NIZK\text{-}PK}_3$ *et* $\mathsf{NIZK\text{-}PK}_4$ *et deux adversaires $\mathcal{A}_5$ et $\mathcal{A}_6$ contre la solidité des schémas NIZK-PK* $\mathsf{NIZK\text{-}PK}_1, \mathsf{NIZK\text{-}PK}_3$*, tel que :*

$$\epsilon \leq N_{\mathsf{Test}}(\mathsf{Adv}^{\mathsf{IND-CCA}}_{\mathcal{A}_1} + \mathsf{Adv}^{\mathsf{sec}}_{\mathcal{A}_2}) + nN_{\mathsf{Test}} \sum_{i=1}^{2} \mathsf{Adv}^{\mathsf{ZK}}_{\mathcal{A}_i} + nN_{\mathsf{Test}}\ell mc\mathsf{Adv}^{\mathsf{Snd}}_{\mathcal{A}_3} + nN_{\mathsf{Test}}\mathsf{Adv}^{\mathsf{Snd}}_{\mathcal{A}_4}.$$

Ébauche de preuve. Dans la suite, nous décrivons la réduction de sécurité pour chaque cas.

Conformité. La conformité du protocole s'appuie sur la conformité des processus d'initialisation, d'enregistrement, d'achat et de récupération de contenu. Dans la réduction, nous excluons en premier la possibilité que le certificat d'une clef échoue à l'étape de vérification, même si la clé a été honnêtement certifiée. En effet, si une telle clef existe, alors une certification (signature) n'est pas vérifiable, malgré le fait qu'elle ait été signée avec une clef légitime. Ici, nous perdons le terme $\mathsf{Adv}^{\mathsf{conforme}}_{\mathcal{A}_2}$ contre la conformité du schéma de certification. En deuxième, nous excluons la possibilité que durant la phase d'achat une signature honnête échoue à la vérification du vendeur. Ce résultat fait perdre $\mathsf{Adv}^{\mathsf{conforme}}_{\mathcal{A}_1}$ contre la conformité du schéma de signature de groupe. Durant la préparation des contenus, nous faisons perdre $\mathsf{Adv}^{\mathsf{conforme}}_{\mathcal{A}_3}$ en excluant la possibilité qu'une preuve NIZK générée durant cette phase ne soit pas vérifiée. Les deux dernières étapes sont nécessaires pour exclure les possibilités qu'un contenu symétriquement chiffré ne déchiffre pas le contenu correct et que tous les tours d'OT échouent. Dans ces deux cas, la réduction cherche à deviner dans quelle cellule de quel WORM (sur un total de ℓcm cellules) insérer la clef symétrique dans ce jeu et respectivement à quel tour d'OT (sur un total de $N_{\mathsf{Test}}m$ tours) le processus échoue, d'où les facteurs supplémentaires dans la déclaration de sécurité.

Non-chaînabilité des acheteurs. Pour cette réduction, nous excluons tout d'abord la probabilité que la certification des clefs des acheteurs révèle des informations sur eux à un adversaire. Comme la réduction doit deviner quels acheteurs injectent le jeu de sécurité pour le canal, nous perdons le terme total $n\mathsf{Adv}^{\mathsf{sec}}_{\mathcal{A}_3}$, dans lequel $\mathcal{A}_3$ est un adversaire contre la sécurité du canal entre CA et l'acheteur. Dans l'étape suivante, nous excluons la possibilité de fuite à travers le canal de communication anonyme entre l'acheteur et le vendeur. De nouveau, comme la réduction doit deviner pour quelle transaction ceci

prend place (ceci donne immédiatement l'indice de l'acheteur), nous perdons le terme $N_{\mathsf{Test}}\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{A}_2}$, dans lequel $\mathcal{A}_2$ est l'avantage de l'adversaire contre l'anonymat du canal de communication anonyme. Il faut noter que, dans la notion de non-chaînabilité des acheteurs, nous incluons la possibilité que le vendeur se comporte mal et choisisse des paramètres malveillants, lui permettant ainsi de distinguer les acheteurs en fonction de leurs empreintes. Cependant, le vendeur doit aussi prouver à l'aide d'une preuve de connaissance à divulgation nulle que les paramètres ont été générés correctement et que les empreintes sont convenablement générées pour chaque contenu préparé. Nous excluons la possibilité que l'adversaire génère des paramètres biaisés par une réduction de la propriété de sûreté de la NIZK-PK utilisant un argument hybride standard (dans lequel nous perdons un facteur ℓ^2, pour le nombre ℓ de contenus que possède le vendeur). Finalement, pour chaque transaction l'acheteur signe un message avec sa clef de signature de groupe. Dans cette dernière réduction, nous éliminons la possibilité que l'adversaire distingue deux acheteurs par leurs signatures, ce qui implique de perdre le terme $\mathsf{Adv}^{\mathsf{full-anon}}_{\mathcal{A}_1}$. Remarquons que nous avons besoin de la propriété d'anonymat complet, car dans la définition de non-chaînabilité forte des acheteurs, l'adversaire est capable de conduire des corruptions arbitraires.

Anti-framing. Dans le jeu anti-framing, l'adversaire peut induire OA à accuser un acheteur innocent de collusion pour créer une copie pirate du contenu. Nous excluons en premier la possibilité de générer de faux paramètres de Tardos et de tricher à la préparation des contenus. Comme pour la preuve précédente, ici, nous perdons un terme $\ell^2\mathsf{Adv}^{\mathsf{Snd}}_{\mathcal{A}_2}$. Ensuite, nous excluons la possibilité que OA accuse faussement un innocent en se basant sur la validité de la preuve (c'est-à-dire, les malfonctions du processus d'accusation). Ici, la réduction doit deviner dans quel oracle de test cela a lieu et quel acheteur est faussement accusé, de là on perd le terme $N_{\mathsf{Test}}n\delta$, dans lequel n est le nombre d'acheteurs enregistrés et δ est la probabilité d'avoir un faux positif à condition que les paramètres de Tardos soient véritablement générés. Une autre façon pour le vendeur de tricher est de générer une preuve accusant un acheteur innocent, malgré le fait que le calcul des scores indique qu'il est innocent. Cette attaque est équivalente à forger une preuve d'accusation, jusqu'à un facteur $N_{\mathsf{Test}}n$ (nous devons deviner dans quelle transaction et pour quel acheteur l'adversaire forge la preuve). Finalement, la traçabilité complète du schéma de signature de groupe assure qu'un acheteur innocent est accusé avec une probabilité d'au plus $\mathsf{Adv}^{\mathsf{full-trace}}_{\mathcal{A}_1}$.

Traçage de traîtres. Cette propriété exige que l'algorithme d'ouverture fournisse en sortie au moins un des membre de la collusion au cas où un contenu aurait été forgé. La preuve de cette propriété est similaire à celle de l'anti-framing. Cependant, cette fois le vendeur n'est pas considéré comme une partie de l'adversaire, par conséquent nous pouvons supposer que les paramètres de Tardos sont bien générés. En cas de collusion, le vendeur calcule les scores de Tardos et transmet un sous-ensemble de signatures, avec une NIZK-PK que l'accusation a été faite correctement à OA. Premièrement, nous excluons la possibilité qu'un adversaire fausse la preuve NIZK-PK ou génère une fausse liste de signatures et de preuves. Cette attaque est équivalente à briser la sûreté de la NIZK-PK, vu que l'adversaire ne connaît pas les paramètres des codes de Tardos (le vendeur est honnête), ce qui fait perdre $N_{\mathsf{Test}}\mathsf{Adv}^{\mathsf{Snd}}_{\mathcal{A}_2}$. Si la liste des signatures est

sincèrement calculée par le vendeur alors la traçabilité complète du schéma de signature de groupe assure que le signataire correct est extrait.
Non-chaînabilité des contenus. Dans PIMENTO+, un acheteur choisit quel contenu acheter, mais contrairement à PIMENTO il envoie directement la signature à OA, qui la certifie. Comme le contenu choisi n'est pas signé, la signature n'apporte pas d'information sur cette valeur. Dans cette réduction, nous excluons d'abord la possibilité que l'adversaire apprenne quelque chose sur le contenu choisi provenant du chiffrement envoyé à OA. Vu que cela est fait pour tout appel à l'oracle de test nous perdons le terme total de $N_{\mathsf{Test}}\mathsf{Adv}^{\mathsf{IND-CCA}}_{\mathcal{A}_1}$. Comme le vendeur est considéré comme étant malveillant, nous excluons la possibilité que le vendeur génère les paramètres des codes de Tardos d'une manière inappropriée, qui conduirait OA à ouvrir les signatures d'un acheteur honnête. Comme pour les preuves précédentes, nous perdons le terme $n\ell\mathsf{Adv}^{\mathsf{ZK}}_{\mathcal{A}_1}$ vu que la réduction doit deviner pour quel acheteur et pour quel contenu cela arrive. L'adversaire peut aussi convaincre OA d'accuser une partie innocente en forgeant une NIZK-PK sur les scores d'accusation pointant un acheteur particulier. Nous éliminons la possibilité que la NIZK-PK à chaque tour du protocole de transfert équivoque révèle une information sur ce témoin (perdant un terme $nN_{\mathsf{Test}}\ell mc\mathsf{Adv}^{\mathsf{Snd}}_{\mathcal{A}_3}$). Ensuite, nous excluons la possibilité que la NIZK-PK envoyée par l'acheteur à la fin de l'exécution du protocole divulgue des informations sur le contenu acheté (on perd le terme $nN_{\mathsf{Test}}\mathsf{Adv}^{\mathsf{Snd}}_{\mathcal{A}_4}$). Finalement, nous excluons la possibilité que le protocole de transfert équivoque lui-même laisse fuir des informations sur le choix de l'entrée, ce qui implique que l'on perd le terme $N_{\mathsf{Test}}\mathsf{Adv}^{\mathsf{sec}}_{\mathcal{A}_2}$.

□

Nous discutons brièvement ici pourquoi notre protocole n'est pas sensible à l'attaque « arrêt et redémarrage » introduite par Kiayias, Leonardos, Lipmaa, Pavlyk et Tang dans [73]. Pour rappel, cette attaque consiste à effectuer une séquence arrêt et redémarrage de la part d'un acheteur malveillant dont l'objectif est de l'aider à apprendre différentes versions des blocs avant l'arrêt complet du protocole. Les conséquences pratiques d'une telle attaque sont les mêmes que celles d'une collusion avec, comme seule différence, que l'attaque est menée par un seul acheteur malveillant. Dans notre protocole, l'entrelacement des deux OT pour délivrer et révéler le *halfword* dissuade un acheteur de réaliser cette attaque. En effet, plus le nombre de blocs qu'il est capable de récupérer est grand, plus grande sera la partie de l'empreinte révélée au vendeur.

3.5. Implémentation

Notre construction montre qu'il est possible d'obtenir – en combinant plusieurs primitives cryptographiques en boîte noire – un protocole de marquage de document assurant de fortes propriétés de sécurité et de protection de la vie privée. Dans cette section, nous discutons de l'aspect implémentation de notre protocole et des blocs de construction.

Les expérimentations reportées dans cette section ont été exécutées sur un Intel i7-2600QM avec 2.40GHz par cœur (un seul cœur du processeur est utilisé) et 4Go de

RAM, avec Mac OSX Yosemite 64bits comme système d'exploitation. Tous les blocs de construction ont été implémentés en Java et sont basés soit sur des librairies existantes soit sur des programmes que nous avons développés. Bien que nous sommes conscients que les temps d'exécution peuvent être améliorés en changeant de langage de programmation (par exemple, en C), notre objectif principal est de montrer que PIMENTO est suffisamment pratique pour être implémenté en tant que preuve de concept. Plus précisément, nous avons implémenté tous les blocs de construction utilisés dans PIMENTO à l'exception du NIZKs et le marquage du document. En effet, bien que les preuves NIZKs puissent être construites pour tout langage NP [65], nous n'avons pu trouver une implémentation open source de celles qui sont nécessaires dans nos protocoles. Dans la suite, nous pointons explicitement les sous-protocoles qui ont été testés.

Pour chaque bloc de construction, nous mesurons le temps d'exécution et l'écart type (exprimés en millisecondes) moyennant 40 tests indépendants, avec comme exception le transfert équivoque où nous moyennons 100 expérimentations. Pour le schéma de signature de groupe, la génération du WORM et de la matrice de clefs, ainsi que pour le calcul des scores, la valeur de m, c'est-à-dire le nombre de blocs, varient dans l'ensemble $m \in \{5000, 10000, 15000, 20000, 25000\}$, alors que N est fixé à 20. Les différentes valeurs considérées pour la taille maximum de la collusion c sont, respectivement par rapport au m ci-dessus : $c \in \{6, 9, 10, 12, 13\}$

Pour le protocole $\mathsf{OT}_1^{\ell N}$, nous avons testé différentes valeurs de ℓN, dans l'ensemble $\in \{20, 200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000\}$. Plus précisément, pour PIMENTO N est fixé à 20 par défaut comme le vendeur connaît l'identité du contenu acheté (donc pour chaque tour, l'entrée du protocole de transfert équivoque est composé de N tuples, alors que pour PIMENTO+ la valeur de N change selon le nombre de contenus (ℓ) dans la base de données du vendeur. À chaque tour de transfert équivoque, la longueur d'une entrée (pour les ℓN possibilités) correspond à la taille de la clef de chiffrement (256 bits) ajouté à la taille du nonce (32bits) ainsi qu'à la taille de l'index de la cellule du WORM (16 bits), qui donne comme taille totale 304 bits. Par exemple, quand $\ell N = 400$ cela veut dire que le vendeur possède une base de données de 20 contenus ($N = 20$). Pour le protocole OT_1^2, qui permet au vendeur de retrouver le *halfword*, nous prenons $\ell N = 2$.

Les résultats obtenus pour les différentes expérimentations sont résumés dans les tableaux 3.1, 3.2 et 3.3. Nous discutons ceux-ci en détail ci-après, en nous concentrant en particulier sur le coût de chaque phase.

Initialisation. Cette phase n'est pas vraiment coûteuse, car elle consiste à générer les paramètres du schéma de signature de groupe, ainsi que quelques paires de clefs publique/privée. Nous testons la phase d'initialisation du schéma dont l'implémentation est décrite dans [16]. Cette implémentation, qui est basée sur les courbes elliptiques, peut être téléchargée à l'adresse suivante : `https://code.google.com/p/group-signature-java/` [28]. Le résultat de l'exécution est présenté dans le tableau 3.1.

28. Accès le 16 février 2016

Phase	TE (ET)
Initialisation GS	1573 (814)
Enregistrement	203.44 (139.16)
Sign	435.73 (34.26)
Vérification de signature	929.88 (47.10)
Ouverture	51.08 (5.26)
OT_1^2	5.72 (22.51)

Tableau 3.1.: Temps d'exécution (TE) et écart type (ET) pour différentes phases de PIMENTO (en millisecondes).

ℓN	TE (ET)
20	26.13 (9.78)
200	196.96 (5.86)
400	380.58 (8.93)
600	564.88 (11.36)
800	749.09 (11.36)
1000	933.48 (17.80)
1200	1117.73 (15.20)
1400	1303.72 (19.58)
1600	1483.53 (20.53)
1800	1733.77 (59.97)
2000	1984.6 (22.69)

Tableau 3.2.: Temps d'exécution (TE) et écart type (ET) pour différentes valeurs de ℓN dans l'$\mathsf{OT}_1^{\ell N}$ (en millisecondes). Comme attendu, l'expérimentation montre que le coût augmente de façon linéaire (Figure. 3.11) avec ℓN.

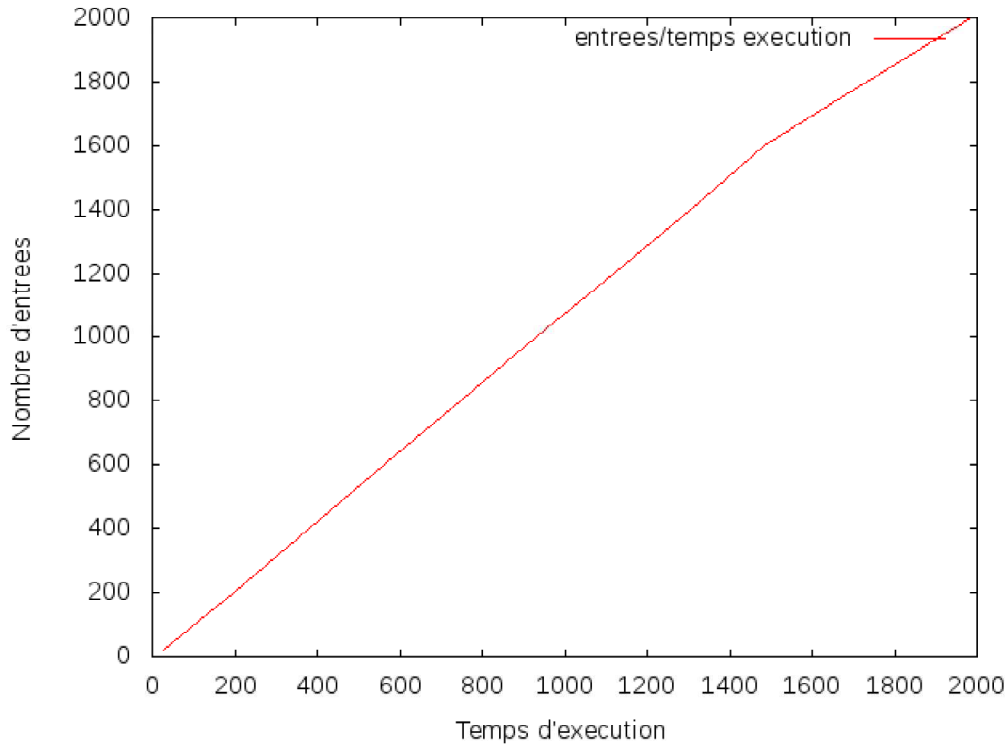

FIGURE 3.11.: Courbe montrant l'évolution du temps d'exécution (en millisecondes, axe x) de la réalisation d'un transfert équivoque pour un nombre donné d'entrées (axe y)

Enregistrement des acheteurs. Cette phase est relativement peu coûteuse. En effet, l'enregistrement d'un acheteur dans le système consiste à exécuter l'algorithme d'enregistrement du schéma de signature de groupe qui est un calcul biparti entre CA et l'acheteur. Nous utilisons la même implémentation que pour la phase d'initialisation pour laquelle nous n'avons testé, ici, que l'algorithme d'enregistrement. La moyenne des temps d'exécution de cette phase est donnée dans le tableau 3.1.

Préparation des contenus. Cette phase est la plus coûteuse du protocole, comme elle consiste à la génération de chaque contenu, Nm bits de marquage et Nm clefs de chiffrement symétrique, à la réalisation de $2m$ fois l'insertion des bits de marquage dans les blocs du contenu, avant d'exécuter Nm fois le processus de chiffrement symétrique. De plus, une preuve NIZK est calculée et est utilisée pour prouver la conformité de ce processus. Bien que cette phase soit très coûteuse, elle n'est faite qu'une fois par contenu présent dans la base de données du vendeur, avant les interactions entre les acheteurs et le vendeur. Par conséquent, plus ce paramètre est grand, plus cette phase est longue, mais le système pourra supporter un plus grand nombre d'acheteurs. De plus, cette phase doit être répétée pour chaque contenu, et uniquement ceux-là, ajouté à la base de données. Les résultats obtenus sont présents dans le tableau 3.3. Comme mentionné précédemment, nous n'incluons pas le coût de la génération des preuves à divulgation nulle de connaissance non interactive dans la discussion ci-dessous et dans les résultats ainsi que le temps de marquage des blocs.

Les paramètres des codes de Tardos (c'est-à-dire, c, m, ϵ et Z) ont été générés par notre implémentation en Java. Le tableau 3.3 résume la moyenne du temps d'exécution et l'écart type obtenus pour différentes valeurs de m.

Plus précisément, pour la génération des clefs de chiffrement, nous avons construit une $N \times m$ matrice contenant des clefs AES de taille 256 bits pour plusieurs valeurs de m en utilisant la librairie BouncyCastle [29].

Pour implémenter la génération du WORM, nous supposons que les contenus du vendeur sont déjà découpés en m blocs marqués. Pour chaque test, nous calculons le temps requis pour chiffrer tous les blocs dans une $N \times m$ matrice en utilisant AES. Pour réaliser cela, nous utilisons un contenu de 1,5Go découpé en m blocs. Cela implique que plus le nombre de blocs est élevé, plus la taille des blocs est petite. Les valeurs possibles de m sont les mêmes que celles utilisées durant la génération de la matrice de clefs AES.

Achat. La phase d'achat consiste à faire m transfert équivoque de 1-parmi-N secrets. Chaque secret correspond à une clef de chiffrement nécessaire au déchiffrement d'une entrée du WORM. Le coût de cette primitive peut être prohibitif si le nombre d'entrées du vendeur est grand. De plus, le protocole *Tout-ou-Rien* conçu par Stern [121], qui est sûr même contre un vendeur qui génère des entrées malveillantes, a une complexité de $O(y2^{\log N})$, où y est la longueur des entrées (dans notre cas une clef et le nonce aléatoire) et N qui est le nombre total d'entrées par colonne du WORM.

Pour la phase d'achat, nous avons testé les algorithmes de signature et de vérification

29. `https://www.bouncycastle.org/fr/` – Accès le 7 mars 2016

	InitTardos	GenClefsAES	GenWORM
m	TE (ET)	TE (ET)	TE (ET)
5000	168 (7)	3181.70 (140.64)	398160.38 (2818.24)
10 000	355 (64)	6301.66 (27.64)	432295.06 (9368)
15 000	524 (77)	9458.02 (47.38)	443367.5 (12948.35)
20 000	758 (158)	12651.71 (84.94)	452377.72 (12249.69)
25 000	901 (165)	16014.80 (95.02)	476529.06 (14192.76)

	DecBlock	ScoreHW	SCMarque
m	TE (ET)	TE (ET)	TE (ET)
5000	21425.31 (93.01)	4.84 (2.57)	8.71 (2.3)
10 000	22613.20 (202.92)	10.61 (2.12)	20.24 (3.05)
15 000	23461.03 (742.32)	15.73 (2.52)	28.74 (3.88)
20 000	23841.39 (589.95)	20.41 (2.89)	38.40 (4.79)
25 000	25159.88 (673.18)	27.34 (2.79)	47.51 (4.60)

Tableau 3.3.: Temps d'exécution (TE) et écart type (ET) pour différentes étapes de notre protocole, nommées : InitTardos pour l'initialisation des codes de Tardos (c'est-à-dire, c, m, ϵ and Z), GenClefsAES pour la génération de la matrice de clefs AES, GenWORM pour la génération du WORM, DecBlocs pour le déchiffrement des blocs avec les clefs AES et ScoreHW pour le calcul de score avec le *halfword* ainsi que SCMarque pour le calcul sur la marque complète. Chaque valeur est en millisecondes.

du schéma de signature de groupe en utilisant la même implémentation que pour les phases d'initialisation et d'enregistrement. Dans ce test, la longueur du message signé par l'acheteur et vérifié par le vendeur est de 24 octets. Les résultats sont présentés dans le tableau 3.1.

Le test du protocole OT_1^n, dans lequel $N * \ell$ est tel que ℓ est le nombre de contenus dans la base de données du vendeur. Le nombre de contenus considéré varie dans l'intervalle $\ell \in \{1, 10, 20, \ldots, 100\}$. Le nombre d'entrées du transfert équivoque quand le vendeur possède un seul contenu est $n = N$, alors que si le vendeur a 10 contenus $n = 10N$. Pour le transfert équivoque OT_1^2, notre test pour PIMENTO ne tient pas compte de la propriété de non-chaînabilité des contenus. Nous n'avons pas pu tester le protocole de transfert équivoque pour PIMENTO+ car celui-ci utilise des preuves à divulgation nulle de connaissance non interactive. Les tests de ces deux protocoles de transfert équivoque ont été faits avec celui disponible dans VMCrypt développé par Lior Malka [87]. La solution implémentée est celle de Naor et Pinkas [93]. Les résultats pour les expérimentations du transfert équivoque OT_1^2 sont présentés dans le tableau 3.1 alors que ceux pour le OT_1^n sont présentés dans le tableau 3.2.

La figure 3.12 montre la taille du *halfword* en fonction de m. Comme on peut le voir, en pratique, la taille réelle n'est pas $\frac{m}{2}$ mais elle en est très proche. La variance est faible comparée à m.

Nous avons aussi calculé le temps nécessaire pour un utilisateur de déchiffrer tous les blocs récupérés durant la phase de récupération du contenu. Les résultats, basés sur la

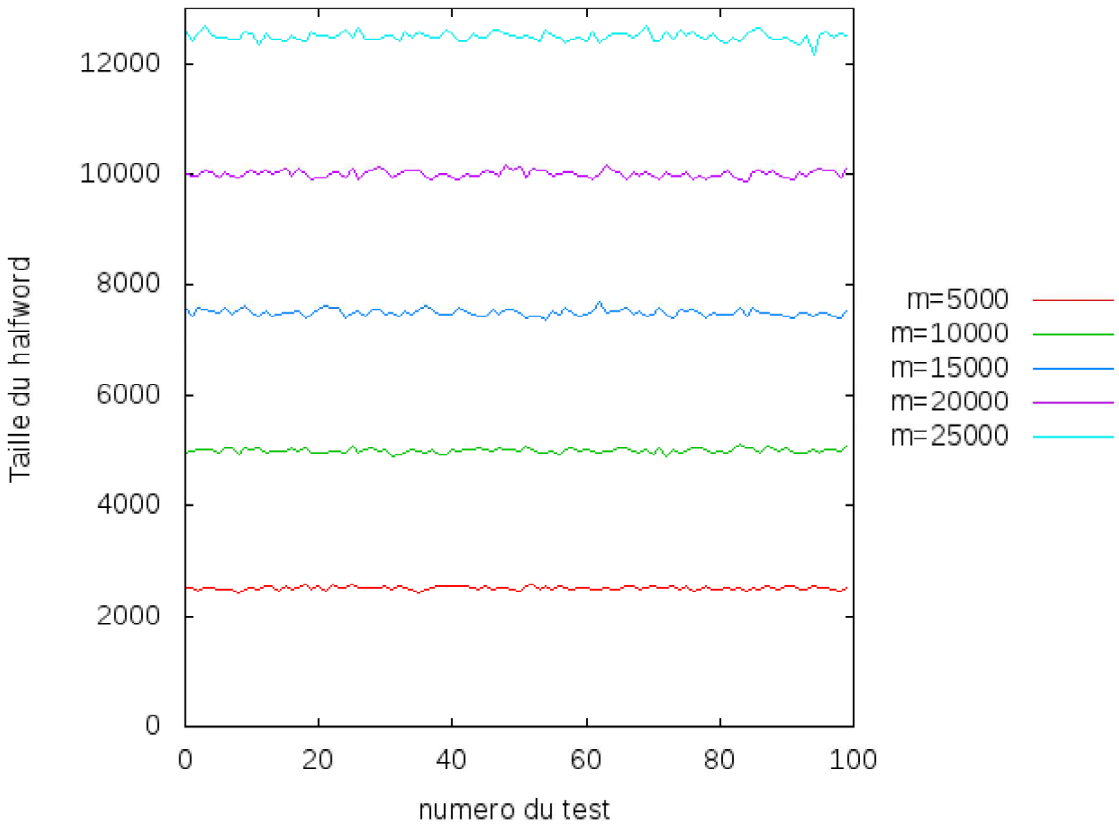

FIGURE 3.12.: Les courbes représentent la taille du *hafword* en fonction de m.

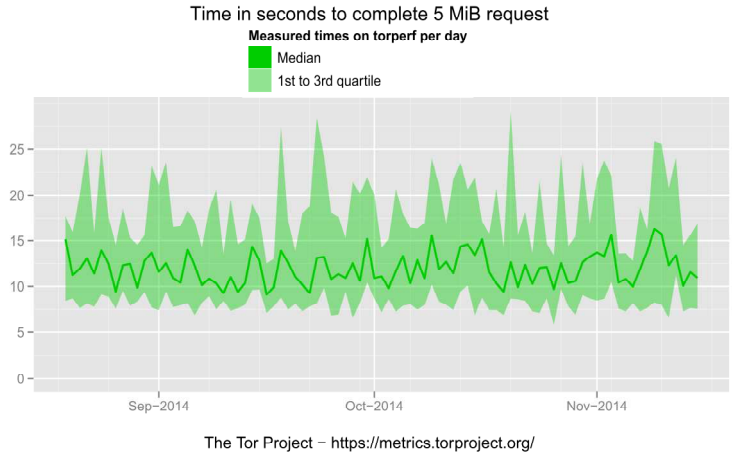

FIGURE 3.13.: Performance de TOR pour un fichier de 5Mo.

librairie cryptographique BouncyCastle, sont disponibles dans le tableau 3.3.

Finalement, notre protocole requiert que toute communication entre le vendeur et les acheteurs se fassent à travers un canal de communication anonyme tel que TOR. Le code source de TOR peut être récupéré sur le site internet du projet [30]. La figure 3.13 [31] montre la performance moyenne de TOR pour un fichier de 5Mo sur une période de 3 mois.

Accusation et ouverture. Les algorithmes d'accusation et d'ouverture ne sont pas exécutés pour chaque achat. En effet, l'algorithme d'accusation est exclusivement exécuté par le vendeur quand des copies forgées sont retrouvées. La phase d'accusation consiste au calcul d'un nombre de scores d'accusation égal au nombre t de transactions pour un contenu spécifique. De plus, une preuve à divulgation nulle de connaissance non interactive est calculée pour démontrer (1) que les scores d'accusation sont supérieurs au seuil d'accusation et (2) que la signature présentée correspond bien à celle reçue pour cette transaction particulière.

Pour tester la phase d'accusation, le temps d'exécution du calcul du score sur le *halfword* et celui sur le calcul du score sur la marque complète ont été mesurés. Les résultats obtenus l'ont été avec notre implémentation des codes de Tardos et sont décrits dans le tableau 3.3. Le temps d'exécution moyen de l'ouverture d'une signature de groupe est présent dans le tableau 3.1 et l'implémentation utilisée est la même que pour les autres algorithmes du schéma de signature de groupe. Dans les tests réalisés, nous considérons que l'empreinte de la copie pirate a été précédemment extraite.

Conclusion. Dans cette section, nous avons décrit les performances obtenues pour différents blocs de constructions utilisés dans l'implémentation de notre protocole

30. `www.torproject.org/download/download` – Accès le 16 février 2016
31. `https://metrics.torproject.org/torperf.html` – Accès le 16 février 2016

comme une preuve de concept. Bien que, notre implémentation de PIMENTO – en particulier la phase d'achat – ne peut pas être, pour le moment, utilisée avec des contraintes de la vie, elle fournit de fortes propriétés de vie privée et de sécurité. De nouvelles avancées en cryptographie pourront améliorer les performances de notre protocole , en terme de temps d'exécution, notamment le protocole de transfert équivoque. Il faut noter que plusieurs blocs de construction de notre protocole ont un temps d'exécution négligeable, tels que le schéma de signature de groupe, la génération des clefs de chiffrement et la phase de déchiffrement. En comparaison, la phase de préparation des contenus (en particulier la génération du WORM) a une grande complexité. Néanmoins, la préparation des contenus est seulement faite une seule fois pour chaque contenu. Pour l'initialisation des codes de Tardos, la génération des clefs, le calcul du score sur le *halfword* et sur la marque complète, la complexité de calcul augmente linéairement en fonction de la taille des entrées alors que le temps nécessaire pour la génération du WORM et la description d'un contenu est constant (pour une taille donnée).

3.6. Discussion sur la conception du protocole

Dans cette section, nous discutons plusieurs choix fondamentaux sur la conception de PIMENTO, soulignant en particulier leurs avantages et inconvénients et des solutions alternatives. Ces considérations ne sont pas spécifiques à notre protocole et sont aussi pertinentes pour la plupart de ceux existants dans la littérature, même si elles sont rarement abordées explicitement.

3.6.1. Calcul du score final et accusation

Comme le vendeur ne doit jamais être capable de distribuer une copie pouvant accuser un acheteur innocent, il est essentiel que le calcul du score final se fasse avec des informations inconnues du vendeur. Pour cette raison, seulement une partie de la marque, appelée *halfword*, est connue du vendeur, et la décision de l'accusation finale doit impliquer l'autre partie de la marque (partie inconnue du vendeur). Par leur nature, les codes de Tardos sont bien adaptés au traçage de traître. En particulier, comme les composants de la marque sont tirés indépendamment, une marque peut facilement être coupée en deux parties (chaque composant est un sous-ensemble de composants, quelles que soient leurs places dans l'empreinte) telles que les scores d'accusation peuvent être indépendamment calculés sur chaque partie. Pour réaliser cela, le vendeur connaît seulement l'indice de chaque composant (pour pouvoir utiliser le bon p_i dans le calcul du score) et adapter chaque seuil à la longueur de la partie utilisée pour le calcul du score (pour garantir que la probabilité de fausse alarme n'excédera pas la valeur maximale autorisée).

La procédure d'accusation que nous proposons pour implémenter PIMENTO est composée de deux étapes.

1. Premièrement, le vendeur identifie un ensemble d'acheteurs suspects en calculant un score partiel basé sur le *halfword* qu'il possède. Si ce score partiel est plus grand que le seuil Z_{half}, alors l'acheteur correspondant est suspecté d'être membre de

la collusion. La première étape de l'accusation est faite n fois si le vendeur ne sait pas quels acheteurs peuvent faire partie de la collusion alors que si le vendeur suspecte déjà des acheteurs particuliers le temps requis peut être significativement réduit.

2. Ensuite, pour chaque acheteur suspect, le vendeur demande à l'OA de calculer un autre score partiel basé sur la partie de la marque non connue par le vendeur. Si ce score est plus grand que le second seuil $Z_{\mathsf{half}'}$, alors l'acheteur est considéré comme coupable. Cette seconde étape requiert que les acheteurs suspectés fournissent leurs empreintes complètes à l'OA. Vu que l'acheteur ne connaît pas quels composants de sa marque sont connus par le vendeur, il n'a pas la possibilité de tricher quand il révèle sa marque sans être détecté avec une forte probabilité. Cette étape ne sera pas fréquemment effectuée comme le nombre d'acheteurs suspects est faible (approximativement la taille de la collusion).

Pour les lecteurs familiers avec les codes de Tardos, ce processus d'accusation est plutôt standard et naturel. Par exemple, Škorić *et co-auteurs* ont montré dans [128] que les scores des membres de la collusion et des utilisateurs innocents suivent tous une distribution Gaussienne (centré en 0 pour le score des innocents), comme illustré par le graphique gauche de la figure 3.14. Cet état est vrai indépendamment de la longueur des mots de codes (pour de mots de codes suffisamment grands) et pour les scores calculés avec un *halfword* connu par le vendeur et pour ceux calculés avec l'autre partie.

Dans notre protocole, l'entrelacement des transferts équivoques durant la phase d'achat entraîne que la longueur des *halfwords* n'est pas toujours la même. Toutefois, comme montré dans la section 3.5, la distribution attendue des longueurs des *halfwords* est centrée en $\frac{m}{2}$, avec un très petit écart type. Par conséquent dans la pratique, les seuils Z_{half1} et Z_{half2} peuvent être fixés comme si les deux parties étaient de longueur égale.

La partie droite de la Figure 3.14 montre la distribution des scores des membres de la collusion et des acheteurs innocents pour les valeurs obtenues avec le *halfword* (connu du vendeur) et l'autre moitié de la marque (inconnu du vendeur). Chaque point (x, y) correspond au score d'un acheteur dont la valeur obtenue sur le *halfword* est égale à x et dont la valeur obtenue sur l'autre partie de la marque est égale à y. Cette figure illustre comment le processus d'accusation se comporte.

Un vendeur malveillant peut aisément générer une marque coïncidant avec les composants du *halfword* et avec la marque d'un acheteur innocent donné. Cependant, cela ne veut pas dire qu'il peut accuser cet acheteur innocent et l'accuser d'être un membre de la collusion. En effet, même si la marque forgée et la marque de l'acheteur innocent coïncident sur les bits du *halfword*, le score partiel qui sera calculé sur la seconde moitié de la marque (qui est inconnue du vendeur) correspondra à un score partiel pour un innocent, comme illustré sur la figure 3.15. Par conséquent, une attaque de framing ne fonctionnera pas avec le processus d'accusation que nous proposons.

L'attaque de retrait de l'accusation introduit dans [73] n'est pas pertinente pour notre protocole. Pour rappel, l'attaque de retrait de l'accusation se présente dans la situation où le vendeur et l'OA ne sont pas d'accord sur l'identité des acheteurs coupables. Avec les codes de Tardos, ceci peut arriver quand l'ensemble des acheteurs coupables

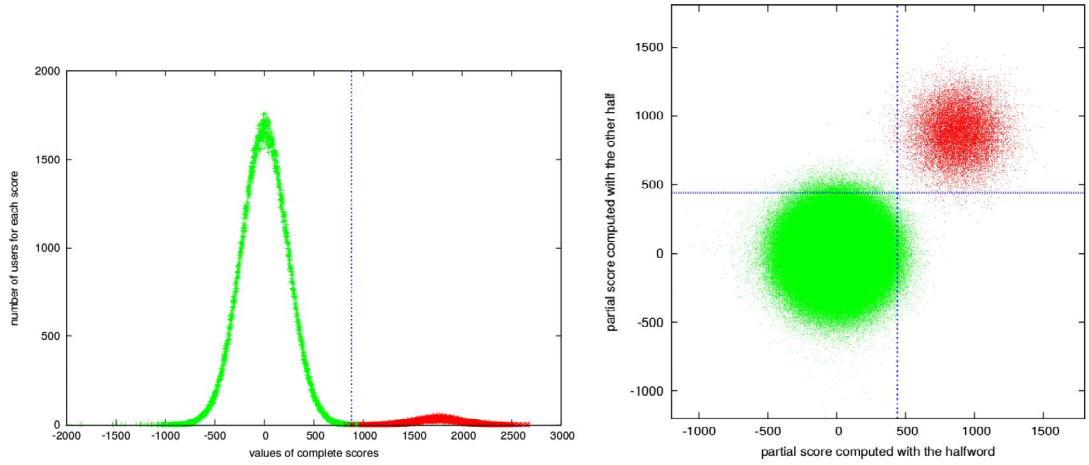

FIGURE 3.14.: Distribution des scores des membres de la collusion (en rouge) et des scores des acheteurs innocents (en vert). Le graphique de droite montre la distribution des scores partiels obtenue avec le *halfword* et l'autre moitié de la marque alors que les lignes bleues correspondent au seuil d'accusation. Le graphique montre les résultats obtenus au travers de 1000 expériences choisies aléatoirement parmi les 10000 faites avec les paramètres $n = 1000$, $c = 20$, $\delta = 10^{-3}$ et $m = 55270$. Ces graphiques sont représentés dans nos résultats.

identifiés par le vendeur et ceux par l'OA n'ont aucun élément en commun. Dans notre solution, comme expliqué précédemment, le vendeur peut fournir un ensemble d'utilisateurs suspects, mais l'OA est le seul à pouvoir calculer les scores d'accusation sur l'empreinte complète. L'attaque par retrait de l'accusation ne peut pas réellement se produire, car la seule accusation importante et légale (scientifiquement parlant) et la plus fiable est celle faite par l'OA. Bien sûr, l'accusation peut échouer, car les codes de Tardos peuvent mener à des faux positifs et des faux négatifs. Cependant, ceci n'est pas dû au protocole de marquage de document lui-même, mais seulement à la logique des codes de Tardos (voir Section 1.6.2).

3.6.2. Génération et échange de la marque/*halfword*

Dans cette section, nous abordons les choix que nous avons faits concernant la génération de la marque et du processus permettant l'envoi du *halfword*.

Dans la première solution, notée $S1$, durant les deux transferts équivoques, la donnée qui est envoyée et reçue est une liste des bits de marquage. Dans ce contexte, l'acheteur reçoit précisément sa marque, qui est un code de Tardos. Ensuite, durant la deuxième étape, le vendeur apprend presque la moitié de ces bits. Ceci implique que l'acheteur connaît sa marque complète et par conséquent qu'une collusion exploitant plusieurs transactions basées sur les mêmes valeurs du secret p_i peut divulguer p_i. Un second inconvénient est que l'acheteur pourra tricher et fournir au vendeur toutes valeurs qu'il

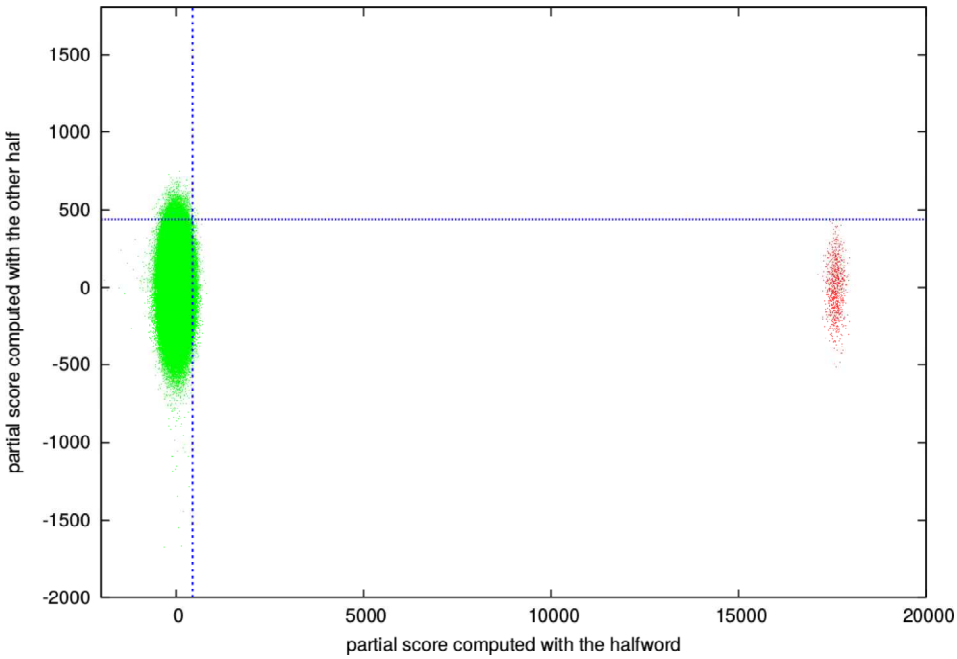

FIGURE 3.15.: Supposons que tous les acheteurs se comportent honnêtement, cette figure montre la distribution des scores obtenus quand un vendeur malveillant essaie d'accuser un acheteur innocent en utilisant la connaissance des *halfword*. Les scores obtenus pour les acheteurs, dont les *halfwords* sont différents de ceux utilisés par le vendeur dans la marque forgée, sont en vert tandis que les scores obtenus pour les acheteurs innocents ciblés que le vendeur voudrait accuser sont en rouge. Ces courbes montrent clairement qu'une telle attaque ne fonctionne pas. Les résultats obtenus sont une moyenne de 1000 expériences, avec les paramètres $n = 1000$, $c = 20$, $\delta = 10^{-3}$ et $m = 55270$.

choisira pour les bits du *halfword* (pas nécessairement ceux obtenus durant le premier transfert équivoque). Si le vendeur veut vérifier que les bits reçus pour le *halfword* sont effectivement tirés en accord avec le secret p_i, un test statistique doit être effectué. À notre connaissance, aucun test n'existe pour le moment dans la littérature.

Finalement, un dernier inconvénient majeur est qu'une étape de plus doit être réalisée pour insérer les bits de la marque dans le contenu. Faire cela sans révéler la marque entière au vendeur ni le contenu original à l'acheteur paraît très coûteux en terme d'exigence de calcul. Pour toutes ces raisons, nous choisissons de ne pas implémenter la méthode $S1$.

Une seconde solution, que nous notons $S2$, peut être de faire le transfert équivoque non plus sur les bits de la marque, mais sur des blocs tatoués. Cette insertion est effectuée comme un pré-calcul du contenu, hors-ligne, et le bit de la marque $\ll 0 \gg$ (respectivement $\ll 1 \gg$) est remplacé par un bloc dans lequel un $\ll 0 \gg$ (respectivement $\ll 1 \gg$) est inséré. Cette solution résout le problème de l'insertion mentionnée dans le paragraphe précédent.

De plus, ceci implique que l'acheteur n'apprend pas sa marque, car il ne connaît pas la bonne clef pour extraire l'information du bloc [32]. La meilleure attaque que peut faire un acheteur malveillant est de faire une collusion avec d'autres acheteurs et d'utiliser plusieurs blocs marqués correspondant au même contenu et à la même valeur secrète p_i pour estimer cette quantité. En faisant cela, il sera seulement capable d'estimer la proportion de deux types de blocs marqués, mais sans réussir à savoir lesquels contiennent un 0 et lesquels contiennent un 1. Ainsi, pour chaque p_i la collusion des acheteurs pourra estimer p_i ou , $1 - p_i$ mais ils ne pourront pas être capables de dire lequel ils ont estimé. Pour empêcher cette attaque, il pourrait être nécessaire d'ajouter un comportement aléatoire dans le processus de tatouage pour que l'insertion du même bit dans le même bloc produise deux blocs différents (qui seront dans le même esprit que la notion de sécurité sémantique utilisée pour le système de chiffrement). Cependant, dans ce cas le vendeur doit générer N versions différentes de chaque bloc du contenu marqué ce qui sera plus coûteux que de générer 2 versions de chaque bloc. Utilisant cette approche, aucun test statistique n'est nécessaire pour convaincre le vendeur que l'acheteur lui fournit des données correctes pour le *halfword*, car le vendeur peut directement vérifier que les blocs qu'il récupère ont bien été choisis parmi ceux qu'il fournit. Cependant, un inconvénient majeur est que le nombre de données transmises entre le vendeur et l'acheteur est vraiment important. Pour cette raison, nous choisissons de ne pas utiliser cette solution dans notre protocole.

Une troisième solution, notée $S3$, est d'utiliser, pour la transaction, ni les bits de la marque ni les blocs marqués du contenu, mais plutôt les clefs utilisées pour chiffrer les blocs marqués du contenu. Cette solution résout le problème de charge utile mentionnée précédemment en plus d'être la plus sûre et la plus pratique des trois solutions. C'est pourquoi nous l'utilisons dans notre protocole.

32. Les techniques de tatouage de document robuste et efficace utilisent la même clef pour insérer et extraire la marque et cette clef ne doit jamais être révélée à un acheteur, car il pourrait l'utiliser pour supprimer complètement la marque

3.7. Conclusion

Les travaux présentés ici sont les premières étapes vers l'intégration de fortes propriétés de protection de la vie privée dans des protocoles de marquage de document. Nos solutions, PIMENTO et PIMENTO+, étendent le protocole de marquage de document asymétrique de Charpentier, Fontaine, Furon et Cox [33] en ajoutant la non-chaînabilité des acheteurs et des articles, tandis que nous préservons une forte probabilité de tracer des traîtres en utilisant les codes de Tardos. De plus, nous avons défini formellement les propriétés de conformité, de non chaînabilité des acheteurs et des articles ainsi que la non-accusation d'acheteur innocent spécifique et de traçage de traître pour PFP-TT. Finalement, nous donnons les limites exactes pour les propriétés de protection de la vie privée et de sécurité de notre protocole.

Le tableau 3.4 résume la comparaison entre PIMENTO et les protocoles présentés dans la section 2.4.

	[106]	[103]	[25]	[113]	[112]	[12]	[49]	[81]	PIMENTO
Anti-Framing	V	V	V	V	V	V	V	V	V
Traitor tracing	(Oui)	(Oui)	(Oui)	(Oui)	(Oui)	Non	(Oui)	(Oui)	Oui
Analyse de sécurité formelle	X	X	V	V	V	X	X	X	V
Anonymat révocable	V	V	V	V	X	V	V	V	V
Non-chaînabilité des acheteurs	X	V	V	V	X	V	X	X	V
Non-chaînabilité des contenus	X	X	X	X	V	V	X	X	V
Compatible avec les codes de Tardos	X	X	X	X	X	X	X	X	V
Implémentation	X	X	X	V	V	X	X	X	P

Tableau 3.4.: Respect des propriétés de sécurité et protection de la vie privée par les protocoles présents dans la littérature. La couleur orange concernant le traçage de traîtres indique, soit que les codes de ces solutions sont trop grands pour être utilisés en pratique, soit que le code est inefficace contre les collusions. En vert, ce sont les codes utilisables en pratique et permettant de contrer des collusions. Le P sur la ligne concernant l'implémentation pour notre protocole PIMENTO signifie que notre implémentation est partiel. En effet, il manque l'implémentation des NIZK-PK pour la compléter.

Conclusion et perspectives

Ces dernières années, la redistribution illégale des contenus protégés par le droit d'auteur est devenue une pratique courante qui permet d'obtenir, par exemple, les dernières séries avant leur sortie en France, de pouvoir visionner des photos, des films sans avoir à payer. Pour dissuader l'utilisation de cette pratique, il existe des protocoles de personnalisation de contenus. Ceux-ci ont pour objectif de dissuader les utilisateurs de partager illégalement des contenus protégés et de garantir au propriétaire du contenu qu'il sera capable de retrouver avec une forte probabilité l'utilisateur malveillant. Cet axe de recherche s'est beaucoup développé ces dernières années, les premiers protocoles asymétriques ayant vu le jour au milieu des années 90.

En parallèle, le besoin de protection de la vie privée s'est également amplifié depuis ces dernières années, car les utilisateurs des technologies de l'information et de la communication ont pris conscience des problèmes de ces derniers. En effet, même sans le savoir, les données personnelles des utilisateurs sont collectées massivement par les sites internet, les entreprises et les états et les utilisateurs commencent à faire attention à leurs données. Du fait de cette collecte massive, il faut proposer des solutions alternatives permettant de préserver la vie privée des individus honnêtes, tout en limitant l'impact que cela peut avoir sur l'utilité ou la performance du service.

Dans la première partie de ce mémoire, nous nous sommes attachés à présenter tout d'abord les concepts fondamentaux de personnalisation de contenu, de traçage de traîtres et de protection de la vie privée. Dans le Chapitre 1, nous avons présenté les concepts de tatouage, code anti-collusion, la personnalisation de contenu avec traçage de traîtres et les attaques contre ces protocoles. Nous avons aussi vu que les codes anti-collusion de Tardos, de par leur nature probabiliste, ne peuvent être utilisés dans les protocoles de la littérature. À ce jour, seulement deux protocoles sont compatibles avec les codes de Tardos : [33] et [73], car ceux-ci ont été conçus pour les utiliser. Dans le chapitre 2, nous avons abordé le concept de protection de la vie privée. Nous avons tout d'abord expliqué ce qu'était la vie privée, avant de présenter le cadre légal de celle-ci et son respect dans la société. Puis, nous avons détaillé les menaces et attaques contre celle-ci avant de présenter les moyens existants pour s'en défendre. Enfin, nous avons présenté les protocoles couplants personnalisation de contenu et protection de la vie privée dans la littérature.

Dans la deuxième partie de ce mémoire, nous avons présenté nos contributions. Le chapitre 3 décrit nos protocoles nommés PIMENTO et PIMENTO+ permettant d'atteindre l'objectif que nous nous étions fixé : réaliser un protocole efficace permettant de personnaliser un contenu numérique dans le but d'effectuer du traçage de traître tout en garantissant la vie privée des utilisateurs. Nous avons aussi prouvé que nous atteignons nos objectifs. Les propriétés garanties sont la non-chaînabilité des acheteurs

et l'anonymat révocable en ce qui concerne le respect de la vie privée ainsi que les propriétés de sécurité d'*anti-framing* et de traçage de traître. Nous avons atteint ces propriétés pour nos deux protocoles. De plus, nous avons également atteint la propriété de non-chaînabilité des contenus numériques pour le protocole PIMENTO. Dans ces deux solutions, nous proposons un modèle de sécurité et des preuves formelles que nous atteignons bien les propriétés spécifiées. Nous avons aussi effectué des tests pratiques montrant qu'il était possible d'implémenter un tel protocole (preuve de concept) bien que ce dernier soit, comme nos tests le montrent, trop coûteux en termes de communications et de calculs. De plus, nous n'avons pas pu tester le protocole entièrement du fait de l'absence d'implémentation des preuves à divulgation nulle de connaissance non interactive (spécifique à notre protocole), ainsi que de l'absence de schéma de tatouage robuste développé en JAVA. Pour finir, dans PIMENTO les contenus numériques sont tout les contenus pouvant être découpés en bloc (pour la génération du WORM) tels que film, chanson et photo. Les bases de données ne peuvent être utilisées sans modification du protocole.

En plus de la personnalisation de contenu comme nous l'avons vu dans la section 2.5, il est possible aussi de personnaliser une base de données comme nous l'avons présentée dans le Chapitre 1. En effet, au vu de la structure de ce type de contenu, les algorithmes de tatouage de contenu plus classique (par exemple, films, chansons) ne sont pas adaptés et la personnalisation ne doit pas modifier les contraintes des requêtes, par exemple statistiques, sur les données. Le traçage de traîtres pour lutter contre la redistribution de bases de données est aussi important et répond à la même problématique que celui pour les contenus numériques décrit dans les Chapitres 1 et 2. C'est pourquoi des outils dédiés aux bases de données ont été étudiés. Nous avons présenté les systèmes principaux existants dans la littérature dans le chapitre 1 et 2. Nous expliquons notre travail en cours, c'est-à-dire la problématique que nous voulons résoudre ainsi que des pistes pour y parvenir ci-après. Résoudre cette problématique peu étudiée dans la littérature, permettrait, par exemple, à un propriétaire de fournir sa base de données médicales à des tiers de façon à ce qu'il puisse tracer les éventuels redistributeurs tout en garantissant aux patients que leurs informations personnelles sont assainies. L'originalité de ce travail vient du fait que nous voulons que la méthode d'assainissement conduise la personnalisation de la base de données. Le défi principal qui nous reste à relever est de trouver un tel mécanisme.

Perspectives

Finalisation et amélioration de nos travaux Tout d'abord, nous voulons trouver une manière de concevoir les NIZK dont nous avons eu besoin dans PIMENTO et de les implémenter. Pour cela, nous allons regarder en détail les preuves de Groth [64] et Groth et Sahai [66]. Pour plus d'informations sur ces preuves, il est recommandé de se référer à la thèse de Paul Lajoie-Mazenc [80]. L'objectif de cette conception de NIZK est de pouvoir diffuser de manière complète le code source de PIMENTO. Mais pour cela, il nous faut aussi trouver une alternative pour le protocole de transfert équivoque.

Il nous faut aussi trouver une alternative à BrokenArrow pour l'algorithme de tatouage pour pouvoir tatouer le mot de code dans des blocs dont la taille dépend de la taille du contenu et de m [33].

De plus, il nous paraît intéressant d'ajouter [16], qui est un protocole de recommandation de contenu préservant la vie privée, dans PIMENTO et dans son extension avec les bases de données pour avoir un protocole complet. En effet, les utilisateurs sont habitués à se faire recommander des contenus (vidéo à la demande, site de commerce électronique, réseaux sociaux, etc). Pour toutes ces situations, utiliser un protocole de recommandation de contenu couplé à un protocole de personnalisation de contenu, tous deux respectant la vie privée, est intéressant pour montrer qu'il est possible d'avoir un système complet fournissant les mêmes propriétés aux utilisateurs qu'un système sans protection de la vie privée.

Aussi, nous voulons terminer notre schéma de personnalisation de bases de données assainies. En effet, nous allons finir ce travail et faire des tests pratiques pour juger l'efficacité réelle de ce dernier.

La question à laquelle nous voulons répondre est la suivante :

Comment faire pour personnaliser une base de données avant sa diffusion, tout en respectant la vie privée des personnes possédant des informations sensibles et personnelles dans cette dernière ?

Bien que cette problématique nous paraisse importante, elle n'a été étudiée dans la littérature qu'à travers deux contributions, [118] et [74], qui utilisent toutes deux le mécanisme d'assainissement implémentant la k-anonymité. En effet, si la base de données comporte des informations personnelles et sensibles sur des individus, alors il est primordial que cette base de données soit assainie (par exemple, en utilisant la confidentialité différentielle) et personnalisée, pour protéger le droit d'auteur, avant sa distribution.

Nous avons identifié quatre moyens de résoudre cette problématique.

1. Il est possible de personnaliser la base de données puis d'utiliser un mécanisme d'assainissement tel que k-anonymité ou l-diversité. Cette solution a comme défaut que l'assainissement pourrait, dans le meilleur des cas, détruire partiellement l'empreinte et, au pire, la détruire complètement. En effet, l'assainissement, dans ce cas, utilise des généralisations et des suppressions qui pourraient supprimer l'empreinte.
2. L'assainissement peut être effectué en premier lieu avant de personnaliser le résultat. Le problème de cette solution vient du fait que l'assainissement, comme dans la solution précédente, utilise des généralisations et des suppressions. Considérant la personnalisation comme une modification de bits d'attributs, cela implique qu'il y aura moins d'attributs qui pourront être marqués (suppression) et que ceux qui le pourront ne le seront potentiellement que sur des bits de poids fort (généralisation).
3. Cette solution consiste à utiliser le mécanisme d'assainissement pour personnaliser la base de données. La personnalisation est en fait un assainissement personnalisé

33. BrokenArrow ne permet de tatouer que des images de taille 512 par 512 pixels.

de la base de données distribuée à chaque utilisateur. Chaque assainissement est proche des autres. L'assainissement peut être, par exemple, du k-anonymat [118] et [74], de la l-diversité ou encore de la confidentialité différentielle. Cette solution n'a pour l'instant pas été étudiée dans la littérature.

4. La dernière manière de faire est d'utiliser de la confidentialité différentielle, obtenue à l'aide d'un mécanisme d'assainissement tel que la (α, β)-privacy [111]. Le but étant tout d'abord d'assainir la base de données avec ce mécanisme, puis de personnaliser le résultat avec un mécanisme de personnalisation de base de données [13, 42].

Les solutions une et deux ne nous paraissent pas optimales, dans le sens où : soit l'utilité des informations contenues dans la base de données est trop dégradée, soit la marque est potentiellement effacée par l'assainissement ce qui rend le traçage de traître beaucoup moins fiable. De plus, la troisième solution n'est pas optimale en terme de protection de la vie privée si celle-ci est réalisée à l'aide, par exemple de k-anonymat ou l-diversité, car ceux-ci sont vulnérables aux attaques par intersection et par homogénéité (voir section 2.3.2). Au contraire, si la confidentialité différentielle est utilisée, alors la protection de la vie privée est meilleure. Il faut dans ce cas trouver un mécanisme paramétrable par un secret, qui conduit la personnalisation tout en répondant aux contraintes de la confidentialité différentielle ainsi qu'aux attaques possibles. Cette dernière possibilité n'a pour le moment pas été étudiée dans la littérature.

Dans la dernière solution, nous pourrions assainir la base de données en utilisant la confidentialité différentielle, avant de marquer le résultat en utilisant l'algorithme de Gross-Amblard *et co-auteurs* [42] qui permet de personnaliser la base de données avant sa diffusion. En effet, dans [42] une clef secrète unique est utilisée pour personnaliser la base de données et rendre chaque copie unique. De plus, cette solution permet de respecter des contraintes d'utilité de la base de données résultante de ces modifications [79].

Dans le futur, nous explorerons en détail les solutions 3 (avec la confidentialité différentielle) et 4. Aussi, nous ne nous intéresserons qu'au traçage de traître et au respect de la vie privée. Plus précisément, contrairement à PIMENTO, PIMENTO+ et notre deuxième contribution, nous ne garantirons pas, dans un premier temps, le respect de la vie privée des acquéreurs. Par conséquent, nous ne respecterons pas les propriétés de non-chaînabilité des contenus, d'anonymat révocable et de non-chaînabilité des acheteurs. Dans un deuxième temps, nous aimerions intégrer notre solution de personnalisation de base de données assainie dans PIMENTO, pour atteindre ces dernières propriétés.

Pour finir, un autre travail intéressant serait de modifier PIMENTO avec comme objectif de remplacer le contenu numérique par une base de données. En d'autres termes, nous voulons essayer d'incorporer la solution de personnalisation de base de données respectant la vie privée dans PIMENTO dans le but de garantir les propriétés de non-chaînabilité des acheteurs et des contenus ainsi que la propriété d'*Anti-framing*. Une adaptation au niveau du protocole PIMENTO nous paraît nécessaire pour cela.

Bibliographie

[1] www.senat.fr/lc/lc33/lc333.html. Accès : 27-07-2015.

[2] atilf.atilf.fr/academie9.htm. Accès : 27-07-2015.

[3] Convention for the protection of human rights and fundamental freedoms. conventions.coe.int/treaty/en/Treaties/Html/005.htm. Accès : 27-07-2015.

[4] Directive 06/24/ec of the european parliament and of the council. eur-lex.europa.eu/legal-content/FR/TXT/HTML/?uri=CELEX:32006L0024&from=EN. Accès : 27-07-2015.

[5] Directive 95/46/ec of the european parliament and of the council. eur-lex.europa.eu/legal-content/EN/TXT/?qid=1438001697662&uri=CELEX:31995L0046. Accès : 27-07-2015.

[6] Obligations of data controllers. ec.europa.eu/justice/data-protection/data-collection/obligations/index_en.htm. Accès : 27-07-2015.

[7] Protecting your data : your rights. ec.europa.eu/justice/data-protection/individuals/rights/index_en.htm. Accès : 27-07-2015.

[8] Déclaration universelle des droits de l'homme de 1948. www.textes.justice.gouv.fr/textes-fondamentaux-10086/droits-de-lhomme-et-libertes-fondamentales-10087/declaration-universelle-des-droits-de-lhomme-de-1948-11038.html, 1948. Accès : 27-07-2015.

[9] L'adresse ip est une donnée à caractère personnel pour l'ensemble des cnil européennes. www.cnil.fr/linstitution/actualite/article/article/ladresse-ip-est-une-donnee-a-caractere-personnel-pour-lensemble-des-cnil-europeennes/, Août 2007. Accès : 27-07-2015.

[10] Compteurs communicants : premières recommandations de la cnil. http://www.cnil.fr/linstitution/actualite/article/article/compteurs-communicants-premieres-recommandations-de-la-cnil/, Avril 2013. Accès : 01-09-2015.

[11] Éléments d'analyse technique du projet de la loi relatif au renseignement. sciences.blogs.liberation.fr/files/265206918-note-interne-de-l-inria.pdf, Avril 2015. Accès : 27-07-2015.

[12] W. Abdul, P. Gaborit, and P. Carré. Private Anonymous Fingerprinting for Color Images in the Wavelet Domain. In *Proceedings of SPIE Multimedia on Mobile Devices*, volume 7542, 2010.

[13] R. Agrawal and J. Kiernan. Watermarking relational databases. In *VLDB*, pages 155–166, 2002.

[14] C. Aguilar Melchor, P. Gaborit, and J. Herranz. Additive homomorphic encryption with t-operand multiplications, 2008. carlos.aguilar@unilim.fr 14140 received 5 Sep 2008, last revised 18 Sep 2008.

[15] B. Aiello, Y. Ishai, and O. Reingold. Priced oblivious transfer : How to sell digital goods. In B. Pfitzmann, editor, *Advances in Cryptology — EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 119–135. Springer Berlin Heidelberg, 2001.

[16] E. Aïmeur, G. Brassard, S. Gambs, and D. Scönfeld. P3ERS : Privacy-Preserving PEer Review System. In *Transactions on Data Privacy*, volume 5(3), pages 553–578, December 2012.

[17] Assemblée Nationale et Sénat Français. Dispositions modifiant la loi du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés. `www.legifrance.gouv.fr/affichTexteArticle.do?idArticle=LEGIARTI000006528061&cidTexte=LEGITEXT000006068624`, Janvier 2013. Accès le 16 février 2016.

[18] E. Bertino, B. Chin Ooi, Y. Yang, and R. Deng. Privacy and ownership preserving of outsourced medical data. In *21st International Conference on Data Engineering, ICDE 2005*, pages 521–532, April 2005.

[19] O. Blayer and T. Tassa. Improved versions of tardos' fingerprinting scheme. *Designs, Codes and Cryptography*, 48(1) :79–103, 2008.

[20] M. Blum, P. Feldman, and S. Micali. Non-Interactive Zero-Knowledge and Its Applications. In *Proceedings of the Annual Symposium on the Theory of Computing (STOC)*, pages 103–112, 1988.

[21] D. Boneh and X. Boyen. Short signatures without random oracles. In *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73. Springer Berlin Heidelberg, 2004.

[22] D. Boneh and J. Shaw. Collusion-secure fingerprinting for digital data. In *Advances in Cryptology - CRYPTO'95*, volume 963 of *Lecture Notes in Computer Science*, pages 452–465. Springer Berlin Heidelberg, 1995.

[23] D. Boneh and J. Shaw. Collusion-secure fingerprinting for digital data. *Information Theory, IEEE Transactions on*, 44(5) :1897–1905, Sep 1998.

[24] J. Brassil, S. Low, N. Maxemchuk, and L. O'Gorman. Electronic marking and identification techniques to discourage document copying. In *INFOCOM '94. Networking for Global Communications., 13th Proceedings IEEE*, pages 1278–1287 vol.3, Jun 1994.

[25] J. Camenisch. Efficient anonymous fingerprinting with group signatures. In *Advances in Cryptology - ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 415–428. Springer Berlin Heidelberg, 2000.

[26] J. Camenisch, G. Neven, and A. Shelat. Simulatable adaptive oblivious transfer. In *Proceedings of the 26th annual international conference on Advances in Cryptology*, EUROCRYPT '07, pages 573–590. Springer-Verlag, 2007.

[27] C. Castelluccia and D. Le Métayer. Loi renseignement : « des dizaines de milliers de personnes vont être suspectées à tort ». `www.lemonde.fr/pixels/article/2015/05/06/loi-renseignement-des-dizaines-de-milliers-de-personnes-vont-etre-suspectees-a-tort_4628392_4408996.html`, Mai 2015. Accès : 27-07-2015.

[28] F. Cayre, C. Fontaine, and T. Furon. *Digital Watermarking : Third International Workshop, IWDW 2004, Seoul, South Korea, October 30 - November 1, 2004, Revised Selected Papers*, chapter Watermarking Attack : Security of WSS Techniques, pages 171–183. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[29] F. Cayre, C. Fontaine, and T. Furon. Watermarking security part I : theory. In E. Delp and P. W. Wong, editors, *Security, Steganography, and Watermarking of Multimedia Contents VII*, volume 5681 of *Proc. SPIE-IS&T Electronic Imaging*, pages 746–757, San Jose, CA, USA, United States, Jan. 2005. SPIE.

[30] F. Cayre, C. Fontaine, and T. Furon. Watermarking security part II : practice. In E. Delp and P. W. Wong, editors, *Security, Steganography, and Watermarking of Multimedia Contents VII*, volume 5681 of *Proc. SPIE-IS&T Electronic Imaging*, pages 758–767, San Jose, CA, USA, United States, Jan. 2005. SPIE.

[31] F. Cérou, T. Furon, and A. Guyader. Experimental Assessment of the Reliability for Watermarking and Fingerprinting Schemes. *EURASIP Journal on Information Security*, 2008 :Article ID 414962, 2008.

[32] A. Charpentier, C. Fontaine, and T. Furon. Décodage em du code de tardos pour le fingerprinting. In *XXIIe colloque GRETSI (traitement du signal et des images), Dijon (FRA), 8-11 septembre 2009*. GRETSI, Groupe d'Etudes du Traitement du Signal et des Images, 2009.

[33] A. Charpentier, C. Fontaine, T. Furon, and I. Cox. An Asymmetric Fingerprinting Scheme based on Tardos Codes. In *Proceedings of the 13-th International Conference on Information Hiding (IH'11)*, pages 43–58. Springer-Verlag, 2011.

[34] A. Charpentier, F. Xie, C. Fontaine, and T. Furon. Expectation maximization decoding of tardos probabilistic fingerprinting code. In *Media Forensics and Security I, part of the IS&T-SPIE Electronic Imaging Symposium, San Jose, CA, USA, January 19, 2009, Proceedings*, 2009.

[35] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2) :84–90, Feb. 1981.

[36] D. Chaum. The dining cryptographers problem : unconditional sender and recipient untraceability. *Journal of Cryptology*, 1(1) :65–75, Mar. 1988.

[37] D. Chaum and E. van Heyst. Group Signatures. In *Advances of Cryptology – EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–265. Springer-Verlag, 1991.

[38] A. Cheddad, J. Condell, K. Curran, and P. Mc Kevitt. Digital image steganography : Survey and analysis of current methods. *Signal Processing*, 90(3) :727–752, 2010.

[39] B. Chor, A. Fiat, and M. Naor. Tracing traitors. In Y. Desmedt, editor, *Advances in Cryptology — CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 257–270. Springer Berlin Heidelberg, 1994.

[40] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private information retrieval. *J. ACM*, 45(6) :965–981, Nov. 1998.

[41] C. Chu and W. Tzeng. Efficient k-out-of-n oblivious transfer schemes with adaptive and non-adaptive queries. In S. Vaudenay, editor, *Public Key Cryptography - PKC 2005*, volume 3386 of *Lecture Notes in Computer Science*, pages 172–183. Springer Berlin Heidelberg, 2005.

[42] C. Constantin, D. Gross-Amblard, and M. Guerrouani. Watermill : An optimized fingerprinting system for highly constrained data. In *Proceedings of the 7th Workshop on Multimedia and Security*, MM&#38 ;Sec '05, pages 143–155, New York, NY, USA, 2005. ACM.

[43] I. J. Cox, M. L. Miller, J. A. Bloom, and C. Honsinger. *Digital watermarking*, volume 1558607145. Springer, 2002.

[44] I. Damgård and M. Jurik. A length-flexible threshold cryptosystem with applications. In R. Safavi-Naini and J. Seberry, editors, *Information Security and Privacy*, volume 2727 of *Lecture Notes in Computer Science*, pages 350–364. Springer Berlin Heidelberg, 2003.

[45] P. de police. En savoir plus sur les cookies. `http://www.prefecturedepolice.interieur.gouv.fr/Footer/Plus-sur-les-cookies`, Novembre 2014. Accès le 01-09-2015.

[46] M. Desoubeaux, C. Herzet, W. Puech, and G. Le Guelvouit. Enhanced blind decoding of tardos codes with new map-based functions. In *15th IEEE International Workshop on Multimedia Signal Processing, MMSP 2013, Pula, Sardinia, Italy, September 30 - Oct. 2, 2013*, pages 283–288, 2013.

[47] R. Dingledine, N. Mathewson, and P. Syverson. Tor : the second-generation onion router. In *Proceedings of the 13th conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 21–21. USENIX Association, 2004.

[48] J. Domingo-Ferrer. A new privacy homomorphism and applications. *Information Processing Letters*, 60(5) :277 – 282, 1996.

[49] J. Domingo-Ferrer. Anonymous fingerprinting based on committed oblivious transfer. In *Public Key Cryptography*, volume 1560 of *Lecture Notes in Computer Science*, pages 43–52. Springer Berlin Heidelberg, 1999.

[50] C. Dwork. Ask a better question, get a better answer a new approach to private data analysis. In T. Schwentick and D. Suciu, editors, *Database Theory – ICDT 2007*, volume 4353 of *Lecture Notes in Computer Science*, pages 18–27. Springer Berlin Heidelberg, 2006.

[51] C. Dwork. Differential privacy : A survey of results. In M. Agrawal, D. Du, Z. Duan, and A. Li, editors, *Theory and Applications of Models of Computation*, volume 4978 of *Lecture Notes in Computer Science*, pages 1–19. Springer Berlin Heidelberg, 2008.

[52] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and D. Chaum, editors, *Advances in Cryptology*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18. Springer Berlin Heidelberg, 1985.

[53] C. Fontaine, S. Gambs, J. Lolive, and C. Onete. Private asymmetric fingerprinting : a protocol with optimal traitor tracing using Tardos codes. In *Third International Conference on Cryptology and Information Security in Latin America (Latincrypt'14)*, Florianopolis, Brazil, Sept. 2014.

[54] T. Furon. Le traçage de traîtres. 2009.

[55] T. Furon and P. Bas. Broken arrows. *EURASIP J. Inf. Secur.*, 2008 :1–13, Jan. 2008.

[56] T. Furon, A. Guyader, and F. Cérou. On the design and optimization of tardos probabilistic fingerprinting codes. In *Information Hiding*, pages 341–356. Springer-Verlag, 2008.

[57] S. Gambs, C. Onete, and J. Robert. Prover Anonymous and Deniable Distance-Bounding Authentication. In *Proceedings of ACM AsiaCCS'14, Accepted for publication*. ACM, 2014.

[58] C. Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 169–178. ACM, 2009.

[59] Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. Protecting data privacy in private information retrieval schemes. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 151–160, New York, NY, USA, 1998. ACM.

[60] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the seventeenth annual ACM symposium on Theory of computing*, STOC '85, pages 291–304. ACM, 1985.

[61] M. Green and S. Hohenberger. Blind identity-based encryption and simulatable oblivious transfer. In *Proceedings of the Advances in Crypotology 13th international conference on Theory and application of cryptology and information security*, ASIACRYPT'07, pages 265–282. Springer-Verlag, 2007.

[62] M. Green and S. Hohenberger. Universally composable adaptive oblivious transfer. In *Advances in Cryptology - ASIACRYPT 2008*, volume 5350 of *Lecture Notes in Computer Science*, pages 179–197. Springer Berlin Heidelberg, 2008.

[63] D. Gross-Amblard. Query-preserving watermarking of relational databases and xml documents. In *PODS*, pages 191–201. ACM, 2003.

[64] J. Groth. Simulation-sound nizk proofs for a practical language and constant size group signatures. In X. Lai and K. Chen, editors, *Advances in Cryptology – ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 444–459. Springer Berlin Heidelberg, 2006.

[65] J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for NP. In *Advances in Cryptology – EUROCRYPT'06*, volume 4004 of *LNCS*, pages 339–358. Springer-Verlag, 2006.

[66] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *Advances in Cryptology EUROCRYPT'08*, volume 4965 of *LNCS*, pages 415–432. Springer-Verlag, 2008.

[67] A. Guellier. Can Homomorphic Cryptography ensure Privacy ? Research Report RR-8568, Inria ; IRISA ; Supélec Rennes, équipe Cidre ; Université de Rennes 1, Oct 2014.

[68] D. J. Hand, H. Mannila, and P. Smyth. *Principles of data mining*. MIT press, 2001.

[69] J. Hermans, A. Pashalidis, F. Vercauteren, and B. Preneel. A new RFID privacy model. In *Proceedings of ESORICS'11*, volume 6879 of *LNCS*, pages 568–587. Springer-Verlag, 2011.

[70] J.-H. Hoepman. Privacy design strategies. In N. Cuppens-Boulahia, F. Cuppens, S. Jajodia, A. Abou El Kalam, and T. Sans, editors, *ICT Systems Security and Privacy Protection*, volume 428 of *IFIP Advances in Information and Communication Technology*, pages 446–459. Springer Berlin Heidelberg, 2014.

[71] S. Katzenbeisser and F. Petitcolas. *Information hiding techniques for steganography and digital watermarking*. Artech house, 2000.

[72] A. Kerckhoffs. La cryptographie militaire. *Journal des sciences militaires*, 9 :5–38, Janvier 1883.

[73] A. Kiayias, N. Leonardos, H. Lipmaa, K. Pavlyk, and Q. Tang. Communication optimal tardos-based asymmetric fingerprinting. In K. Nyberg, editor, *Topics in Cryptology — CT-RSA 2015*, volume 9048 of *Lecture Notes in Computer Science*, pages 469–486. Springer International Publishing, 2015.

[74] P. Kieseberg, S. Schrittwieser, M. Mulazzani, I. Echizen, and E. Weippl. An algorithm for collusion-resistant anonymization and fingerprinting of sensitive microdata. *Electronic Markets*, 24(2) :113–124, 2014.

[75] M. Kohlweiss, U. Maurer, C. Onete, B. Tackmann, and D. Venturi. Anonymity-Preserving Public-Key Encryption : A Constructive Approach. In *Proceedings of Privacy Enhancing Technologies PETS'13*, volume 7981 of *LNCS*, pages 19–39. Springer-Verlag, 2013.

[76] M. Kuribayashi and H. Tanaka. Fingerprinting protocol for images based on additive homomorphic property. *IEEE Transactions on Image Processing*, 14(12) :2129–2139, Dec 2005.

[77] T. Laarhoven and B. de Weger. Optimal symmetric tardos traitor tracing schemes. *Designs, Codes and Cryptography*, 71(1) :83–103, 2014.

[78] T. Laarhoven, J. Doumen, P. Roelse, B. Škorić, and B. de Weger. Dynamic tardos traitor tracing schemes. *CoRR*, abs/1111.3597, 2011.

[79] J. Lafaye, D. Gross-Amblard, C. Constantin, and M. Guerrouani. Watermill : An optimized fingerprinting system for databases under constraints. *Knowledge and Data Engineering, IEEE Transactions on*, 20(4) :532–546, April 2008.

[80] P. Lajoie-Mazenc. In *Réputation et respect de la vie privée dans les réseaux dynamiques et auto-organisés*. 2015.

[81] C.-L. Lei, P.-L. Yu, P.-L. Tsai, and M.-H. Chan. An efficient and anonymous buyer-seller watermarking protocol. *Image Processing, IEEE Transactions on*, 13(12) :1618–1626, Dec 2004.

[82] lemonde.fr. `www.lemonde.fr/vie-en-ligne/article/2015/04/29/l-allemagne-aurait-espionne-des-officiels-francais-et-europeens-pour-le-compte-de-la-nsa_4625388_4409015.html`, Avril 2015. Accès : 01-09-2015.

[83] A. Lindell. Efficient fully-simulatable oblivious transfer. In *Proceedings of the 2008 The Cryptopgraphers' Track at the RSA conference on Topics in cryptology*, CT-RSA'08, pages 52–70, Berlin, Heidelberg, 2008. Springer-Verlag.

[84] H. Lipmaa. An oblivious transfer protocol with log-squared communication. In J. Zhou, J. Lopez, R. H. Deng, , and F. Bao, editors, *Information Security*, volume 3650 of *Lecture Notes in Computer Science*, pages 314–328. Springer Berlin Heidelberg, 2005.

[85] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. L-diversity : Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), Mar. 2007.

[86] F. MacWilliams and N. Sloane. *The theory of error correcting codes*. Elsevier, 1977.

[87] L. Malka. Vmcrypt : modular software architecture for scalable secure computation. In *Proceedings of the 18th ACM conference on Computer and communications security*, CCS '11, pages 715–724, New York, NY, USA, 2011. ACM.

[88] S. Martello and P. Toth. Approximation schemes for the subset-sum problem : Survey and experimental analysis. *European Journal of Operational Research*, 22(1) :56 – 69, 1985.

[89] P. Meerwald and T. Furon. Iterative single tardos decoder with controlled probability of false positive. In *Multimedia and Expo (ICME), 2011 IEEE International Conference on*, pages 1–6, July 2011.

[90] D. Megías and J. Domingo-Ferrer. Privacy-aware peer-to-peer content distribution using automatically recombined fingerprints. *Multimedia Systems*, 20(2) :105–125, 2014.

[91] N. Memon and P. Wong. Protecting digital media content. *Commun. ACM*, 41(7) :34–43, 1998.

[92] N. Memon and P. Wong. A buyer-seller watermarking protocol. *Image Processing, IEEE Transactions on*, 10(4) :643–649, Apr 2001.

[93] M. Naor and B. Pinkas. Efficient oblivious transfer protocols. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, SODA '01, pages 448–457. Society for Industrial and Applied Mathematics, 2001.

[94] A. Narayanan and V. Shmatikov. Robust de-anonymization of large datasets (how to break anonymity of the netflix prize dataset). 2008.

[95] K. Nuida, F. Satoshi, M. Hagiwara, T. Kitagawa, H. Watanabe, K. Ogawa, and H. Imai. An improvement of discrete tardos fingerprinting codes. *Des. Codes Cryptography*, 52(3) :339–362, Sept. 2009.

[96] T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In K. Nyberg, editor, *Advances in Cryptology — EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 308–318. Springer Berlin Heidelberg, 1998.

[97] J. Oosterwijk, B. Škorić, and J. Doumen. Optimal suspicion functions for tardos traitor tracing schemes. In *ACM Information Hiding and Multimedia Security Workshop, IH&MMSec '13, Montpellier, France, June 17-19, 2013*, pages 19–28, 2013.

[98] A. Oprea and K. Bowers. Authentic time-stamps for archival storage. In *Proceedings of the 14th European conference on Research in computer security*, ESORICS'09, pages 136–151. Springer-Verlag, 2009.

[99] R. Ostrovsky and S. William E. A survey of single-database private information retrieval : techniques and applications. In *Proceedings of the 10th international conference on Practice and theory in public-key cryptography*, PKC'07, pages 393–411, Berlin, Heidelberg, 2007. Springer-Verlag.

[100] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of the 17th international conference on Theory and application of cryptographic techniques*, EUROCRYPT'99, pages 223–238. Springer-Verlag, 1999.

[101] C. Peikert, A. shelat, and A. Smith. Lower bounds for collusion-secure fingerprinting. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '03, pages 472–479, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.

[102] M. Peyrat. La publicité ciblée en ligne. `www.cnil.fr/fileadmin/documents/La_CNIL/actualite/Publicite_Ciblee_rapport_VD.pdf`, Février 2009. Accès : 27-07-2015.

[103] B. Pfitzmann and A. Sadeghi. Coin-based anonymous fingerprinting. In *Proceedings of the 17th international conference on Theory and application of cryptographic techniques*, EUROCRYPT'99, pages 150–164, Berlin, Heidelberg, 1999. Springer-Verlag.

[104] B. Pfitzmann and A. Sadeghi. Anonymous fingerprinting with direct non-repudiation. In *Proceedings of the 6th International Conference on the Theory and Application of Cryptology and Information Security : Advances in Cryptology*, ASIACRYPT '00, pages 401–414. Springer-Verlag, 2000.

[105] B. Pfitzmann and M. Schunter. Asymmetric fingerprinting. In *Proceedings of the 15th annual international conference on Theory and application of cryptographic techniques*, EUROCRYPT'96, pages 84–95. Springer-Verlag, 1996.

[106] B. Pfitzmann and M. Waidner. Anonymous fingerprinting. In *Proceedings of the 16th annual international conference on Theory and application of cryptographic techniques*, EUROCRYPT'97, pages 88–102. Springer-Verlag, 1997.

[107] B. Pfitzmann and M. Waidner. Asymmetric fingerprinting for larger collusions. In *Proceedings of the 4th ACM conference on Computer and communications security*, CCS '97, pages 151–160. ACM, 1997.

[108] C. Popescu. Applications of group signatures to anonymous fingerprinting schemes. In *Video/Image Processing and Multimedia Communications 4th EURASIP-IEEE Region 8 International Symposium on VIPromCom*, pages 177–182, 2002.

[109] A. Qureshi, D. Megías, and H. Rifà-Pous. Framework for preserving security and privacy in peer-to-peer content distribution systems. *Expert Systems with Applications*, 42(3) :1391–1408, 2015.

[110] M. Rabin. How to exchange secrets with oblivious transfer. http ://eprint.iacr.org/2005/187, 1981.

[111] V. Rastogi, D. Suciu, and S. Hong. The boundary between privacy and utility in data publishing. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, VLDB '07, pages 531–542. VLDB Endowment, 2007.

[112] A. Rial, J. Balasch, and B. Preneel. A privacy-preserving buyer seller watermarking protocol based on priced oblivious transfer. *Information Forensics and Security, IEEE Transactions on*, 6(1) :202–212, March 2011.

[113] A. Rial, M. Deng, T. Bianchi, A. Piva, and B. Preneel. A provably secure anonymous buyer-seller watermarking protocol. In *IEEE Transactions on Information Forensics and Security*, volume 5, pages 920–9310. IEEE, 2010.

[114] A. Rial, M. Kohlweiss, and B. Preneel. Universally composable adaptive priced oblivious transfer. In *Pairing-Based Cryptography–Pairing 2009*, pages 231–247. Springer, 2009.

[115] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2) :120–126, Feb. 1978.

[116] A. Sadeghi. How to break a semi-anonymous fingerprinting scheme. In *Proceedings of the 4th International Workshop on Information Hiding*, IHW '01, pages 384–394. Springer-Verlag, 2001.

[117] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). In *Proceedings of the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, PODS '98, pages 188–. ACM, 1998.

[118] S. Schrittwieser, P. Kieseberg, I. Echizen, S. Wohlgemuth, N. Sonehara, and E. Weippl. An algorithm for k-anonymity-based fingerprinting. In *Digital Forensics and Watermarking*, volume 7128 of *Lecture Notes in Computer Science*, pages 439–452. Springer Berlin Heidelberg, 2012.

[119] A. Simone and B. Škorić. Accusation probabilities in tardos codes : beyond the gaussian approximation. *Des. Codes Cryptography*, 63(3) :379–412, 2012.

[120] R. Sion, M. Atallah, and S. Prabhakar. Rights protection for relational data. In *SIGMOD Conference*, pages 98–109, 2003.

[121] J. Stern. A new and efficient all-or-nothing disclosure of secrets protocol. In *Advances of Cryptology ASIACRYPT'98*, volume 1514 of *LNCS*, pages 357–371. Springer-Verlag, 1998.

[122] L. Sweeney. Simple demographics often identify people uniquely. `http://ggs685.pbworks.com/w/file/fetch/94376315/Latanya.pdf`, 2000. Accessed : 19-06-2015.

[123] L. Sweeney. K-anonymity : A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5) :557–570, Oct. 2002.

[124] M. Szadkowski and D. Leloup. Prism, snowden, surveillance : 7 questions pour tout comprendre. `www.lemonde.fr/technologies/article/2013/07/02/prism-snowden-surveillance-de-la-nsa-tout-comprendre-en-6-etapes_3437984_651865.html`. Accès : 27-07-2015.

[125] G. Tardos. Optimal probabilistic fingerprint codes. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, STOC '03, pages 116–125. ACM, 2003.

[126] G. Tardos. Optimal probabilistic fingerprint codes. *J. ACM*, 55(2), 2008.

[127] S. Vaudenay. On privacy models for RFID. In *Proceedings of Advances in Cryptology – Asiacrypt'07*, volume 4883 of *LNCS*, pages 68–87. Springer-Verlag, 2007.

[128] B. Škorić, S. Katzenbeisser, and M. Celik. Symmetric tardos fingerprinting codes for arbitrary alphabet sizes. In *Designs, Codes and Cryptography*, pages 137–166. Springer-Verlag, 2008.

[129] B. Škorić and J. Oosterwijk. Binary and q-ary tardos codes, revisited. *Des. Codes Cryptography*, 74(1) :75–111, 2015.

[130] B. Škorić, J. Oosterwijk, and J. Doumen. The holey grail A special score function for non-binary traitor tracing. In *2013 IEEE International Workshop on Information Forensics and Security, WIFS 2013*, pages 180–185, 2013.

[131] B. Škorić, T. Vladimirova, M. Celik, and J. Talstra. Tardos fingerprinting is better than we thought. *CoRR*, abs/cs/0607131, 2006.

[132] N. Wagner, R. Fountain, and R. Hazy. The fingerprinted database. In *Data Engineering, 1990. Proceedings. Sixth International Conference on*, pages 330–336, Feb 1990.

[133] N. R. Wagner. Fingerprinting. In *Proceedings of the 1983 IEEE Symposium on Security and Privacy*, SP '83, pages 18–. IEEE Computer Society, 1983.

[134] F. Xie, C. Fontaine, and T. Furon. Un schéma complet de traçage de documents multimédia reposant sur des versions améliorées des codes de Tardos et de la technique de tatouage " Broken Arrows ". In *Proc. XXIIème colloque du GRETSI*, Dijon, France, Sept. 2009.

[135] L. Yingjiu, V. Swarup, and S. Jajodia. Fingerprinting relational databases : schemes and specialties. *Dependable and Secure Computing, IEEE Transactions on*, 2(1) :34–45, Jan 2005.

Résumé

Depuis la démocratisation d'Internet, la diffusion de contenus numériques protégés par des droits d'auteur (films, photos, ...) s'est accrue de manière importante, en particulier avec les réseaux pairs-à-pair qui permettent de partager des contenus facilement. D'un côté, cette situation pose problème aux ayants droit de ces contenus qui veulent contrôler leur diffusion et dissuader les potentiels contrevenants. De l'autre côté, le traçage constant et généralisé des utilisateurs par des entreprises ou des gouvernements conduit ces utilisateurs à devenir méfiants et provoque une crise de confiance.

L'objectif principal de cette thèse est de pouvoir tracer les utilisateurs malveillants ayant redistribué un contenu illégalement tout en préservant la vie privée des utilisateurs honnêtes. Pour atteindre cet objectif, nous avons conçu un protocole de personnalisation de contenus anonyme qui modifie un contenu avant sa diffusion avec une empreinte unique, tout en garantissant aux utilisateurs honnêtes que leur vie privée est protégée. Cette empreinte permet de tracer des traîtres ayant rediffusé illégalement un contenu et provient d'un code anti-collusion permettant de tracer les traîtres en cas de collusion. Plus précisément, ce protocole est le premier de la littérature à fournir des propriétés fortes de sécurité et de respect de la vie privée se basant sur les codes anti-collusion de Tardos. Les propriétés atteintes par ce protocole sont: traçage de traîtres, anti-framing, anonymat révocable, non-chaînabilité des acheteurs et des contenus distribués. Ce protocole combine différentes briques de constructions telles que le transfert équivoque, les signatures de groupe et un canal de communication anonyme. Il est le premier de la littérature à allier des propriétés fortes d'anonymat avec la performance des codes anti-collusion de Tardos.

Mots-clés : Personnalisation de contenu, Cryptographie, Vie privée, Anonymat, Sécurité informatique

Abstract

Since the democratization of Internet, the illegal redistribution of copyrighted content (movie, picture, ...) has grown tremendously, in particular to the possibility of sharing easily this type of content on P2P network. On the one hand, this situation is problematic for the owner of the content as they want to control the distribution of such data and deter potential offenders. On the other hand, the constant and generalizing tracking of users by companies and governments has lead to a trust crisis.

The main objective of this thesis is to be able to trace malicious users who redistribute illegally a content while protecting their privacy if they behave honestly. To reach this objective, we have designed an anonymous fingerprinting protocol enabling the personalization of a content before his distribution with a unique codeword. This codeword can be used to trace malicious users of they redistribute illegally a content. The codeword comes from an anti-collusion Tardos code that allow to trace malicious users even if the forgery is obtained by a collusion.
This protocol reaches strong security and privacy properties such as traitor tracing, anti-framing, revocable anonymity, buyer and item unlinkability. To achieve these properties, the protocol combines building block such as oblivious transfer, group signatures and an anonymous communication channel. This protocol is the first one in the literature to reach strong anonymity properties as well as the performance of Tardos anti-collusion codes.

Keywords : Content personnalisation, Privacy, Anonymity, Cryptography, Information security

TELECOM Bretagne

n° d'ordre : 2016telb0400
Télécom Bretagne
Technopôle Brest-Iroise - CS 83818 - 29238 Brest Cedex 3
Tél : + 33(0) 29 00 11 11 - Fax : + 33(0) 29 00 10 00